# AutoBreaking Ultimate Full Book

Design, Architecture, Code and Traceability

Generated on 2026-08-13 at 09:59

**Comprehensive Engineering Reference**

## Table of Contents

AutoBreaking Ultimate Full Book

- Quick Navigation
- Meta

- Architecture Narrative
- Module Coupling Graph
- Dependency Tree Snapshot
- Working Tree Status
- Master File Catalog
- File Deep Dive: .github/pull_request_template.md
    - Symbol Index
    - Content Snapshot
- Problem
- Changes
- How to test
- Artifacts
- Acceptance checklist
- File Deep Dive: .github/workflows/ci.yml
    - Symbol Index
    - Line by Line Change Impact
- File Deep Dive: .github/workflows/fuzz-smoke.yml
    - Symbol Index
    - Line by Line Change Impact
- File Deep Dive: .github/workflows/integration-mock.yml
    - Symbol Index
    - Line by Line Change Impact
- File Deep Dive: .github/workflows/nightly-montecarlo.yml
    - Symbol Index
    - Line by Line Change Impact
- File Deep Dive: AUDIT_FULL_2026-08-12.md
    - Symbol Index
    - Content Snapshot

Full Technical Audit - AutoBreaking

- 1. What was executed
- 2. Execution evidence summary
- 3. Severity summary
- 4. Confirmed findings (highest priority)
    - High
    - Medium
- 5. Agent-flagged findings requiring targeted confirmation
- 6. Module-by-module status

7. API and integration audit conclusions
8. Recommended remediation roadmap
9. Final audit verdict
File Deep Dive: CHANGELOG.md
Symbol Index
Content Snapshot
Changelog
[Unreleased]
Added
Changed
File Deep Dive: CODE_EXAMPLES.md
Symbol Index
Content Snapshot
■ Exemplos de Código - Sistema de Freio com ABS
1. Exemplo: Modificar Frequência ABS
Localização
Código Original
Modificação: Aumentar para 10 Hz
Recompilar e Testar
Selecione Emergency Brake (tecla 2)
Observe ciclos ABS acontecerem ~25% mais rápido
2. Exemplo: Ajustar Limiar de Travamento
Localização
Código Original
Modificação: Mais Sensível (4 km/h)
Impacto
3. Exemplo: Mudar Pressão de Liberação ABS
Localização
Código Original
Modificação: Mais Agressivo
Modificação: Mais Conservador
4. Exemplo: Adicionar Novo Cenário
Passo 1: Adicionar Variante de Enum
Passo 2: Implementar Display
Passo 3: Implementar Lógica
Passo 4: Adicionar Tecla de Ativação
Passo 5: Atualizar Controles (comentário)

- 5. Exemplo: Implementar Função de Análise
  - Novo Módulo: metrics.rs
- File Deep Dive: Cargo.toml
  - Symbol Index
  - Line by Line Change Impact
- File Deep Dive: J1939_CAN_MANUAL.md
  - Symbol Index
  - Content Snapshot

Manual Completo: CAN Bus e J1939

- Guia Definitivo para Veículos Pesados e Maquinário Agrícola

PARTE 1: CAN BUS — FUNDAMENTOS COMPLETOS

- 1.1 O que é o CAN Bus
  - Por que CAN e não Ethernet ou RS-485?
- 1.2 Camada Física (Physical Layer) — ISO 11898-2
  - Diferencial e Topologia
  - Níveis de Tensão
  - Terminadores
  - Velocidades e Comprimentos Máximos
- 1.3 Formato do Frame CAN 2.0
  - Frame de Dados (Data Frame)
  - Bits por Frame (CAN 2.0B, 8 bytes dados, sem bit stuffing)
- 1.4 Arbitração — Como Múltiplos Nós Compartilham o Barramento
- 1.5 Tipos de Frames CAN
  - Data Frame
  - Remote Frame (RTR=1)
  - Error Frame
  - Overload Frame
- 1.6 Detecção e Confinamento de Erros
  - Tipos de Erro Detectados
  - Contadores de Erro (ISO 11898-1)
  - Estados do Nó (Error Confinement)
- 1.7 Bit Stuffing
- File Deep Dive: PROJECT_SUMMARY.md
  - Symbol Index
  - Content Snapshot

■ Sumário Completo do Projeto - Sistema de Freio Autônomo com ABS

- ■ O que você tem agora

- Estrutura de Arquivos
- Documentação Incluída
    1. **README.md** (este arquivo)
    2. **TESTING.md**
    3. **ANALYSIS.md**
    4. **TUTORIALS.md**
    5. **ARCHITECTURE.md**
- Recursos Principais
    - Sistema de Simulação (lib.rs)
    - Interface Visual (main.rs)
- Como Começar
    1. Compilação Inicial
    2. Execução
    3. Testar Cenários
- Características Técnicas
    - Performance
    - Física
    - Fidelidade
- Dados Exportáveis
- Casos de Uso
    - Educacional
    - Pesquisa
    - Protótipo
- Tecnologias Utilizadas
    - Linguagem
    - Dependências

# AutoBreaking Ultimate Full Book

> Complete design-book and code-book generated from repository state with maximal practical coverage.

## Quick Navigation

- Meta
- Architecture Narrative
- Module Coupling Graph
- Dependency Tree Snapshot

- Master File Catalog
- Per-file Deep Dive
- Cross File Traceability
- Change Simulation Playbook

## Meta

- Generated at: 2026-08-13 09:55:32
- Git commit: 237eebc
- Files covered: 109
- Rust files covered: 62

## Architecture Narrative

1. Core simulation and ECU domain logic live under src and feed deterministic/runtime scenarios.
2. Live hardware and flashing orchestration live in src/io with strict protocol checks and gating.
3. Binary entrypoints in src/bin drive CLI workflows and template bridge interactions.
4. tests contains regression, property, and integration suites including vendor bridge e2e.
5. scripts and .github files define operational and CI automation.

## Module Coupling Graph

```
graph LR
  adas --> sensors
  boot_sequence --> can_gateway
  boot_sequence --> j1939
  can_gateway --> j1939
  can_network --> j1939
  ecm_heavy --> j1939
  ecm_mock_provider --> io
  ecu_abs --> j1939
  ecu_bcm --> j1939
  ecu_ecm --> j1939
  ecu_hcm --> j1939
  ecu_icm --> j1939
  ecu_tcm --> j1939
  ecu_vcm --> j1939
  electrical --> IgnitionState
  implement --> j1939
  io_allowlist --> io
  io_live_runner --> io
  io_production_program --> HeavyMachinery
  sensors --> vehicle
  uds --> j1939
```

## Dependency Tree Snapshot

<details>

<summary>Expand full dependency tree</summary>

```
auto_breaking v0.3.0 (C:\Users\costabr\OneDrive - Caterpillar\Documents\Rust\AutoBreaking)
|-- chrono v0.4.45
|   |-- num-traits v0.2.19
|   |   [build-dependencies]
|   |   `-- autocfg v1.5.1
```

```
|   `-- windows-link v0.2.1
|-- crc32fast v1.5.0
|   `-- cfg-if v1.0.4
|-- csv v1.4.0
|   |-- csv-core v0.1.13
|   |   `-- memchr v2.8.3
|   |-- itoa v1.0.18
|   |-- ryu v1.0.23
|   `-- serde_core v1.0.229
|-- eframe v0.29.1
|   |-- ahash v0.8.12
|   |   |-- cfg-if v1.0.4
|   |   |-- once_cell v1.21.4
|   |   `-- zerocopy v0.8.56
|   |       `-- zerocopy-derive v0.8.56 (proc-macro)
|   |           |-- proc-macro2 v1.0.107
|   |           |   `-- unicode-ident v1.0.24
|   |           |-- quote v1.0.47
|   |           |   `-- proc-macro2 v1.0.107 (*)
|   |           `-- syn v2.0.119
|   |               |-- proc-macro2 v1.0.107 (*)
|   |               |-- quote v1.0.47 (*)
|   |               `-- unicode-ident v1.0.24
|   |   [build-dependencies]
|   |   `-- version_check v0.9.5
|   |-- document-features v0.2.12 (proc-macro)
|   |   `-- litrs v1.0.0
|   |-- egui v0.29.1
|   |   |-- accesskit v0.16.3
|   |   |-- ahash v0.8.12 (*)
|   |   |-- emath v0.29.1
|   |   |   `-- bytemuck v1.25.2
|   |   |       `-- bytemuck_derive v1.12.0 (proc-macro)
|   |   |           |-- proc-macro2 v1.0.107 (*)
|   |   |           |-- quote v1.0.47 (*)
|   |   |           `-- syn v3.0.3
|   |   |               |-- proc-macro2 v1.0.107 (*)
|   |   |               |-- quote v1.0.47 (*)
|   |   |               `-- unicode-ident v1.0.24
|   |   |-- epaint v0.29.1
|   |   |   |-- ab_glyph v0.2.32
|   |   |   |   |-- ab_glyph_rasterizer v0.1.10
|   |   |   |   `-- owned_ttf_parser v0.25.1
|   |   |   |       `-- ttf-parser v0.25.1
|   |   |   |-- ahash v0.8.12 (*)
|   |   |   |-- bytemuck v1.25.2 (*)
|   |   |   |-- ecolor v0.29.1
|   |   |   |   |-- bytemuck v1.25.2 (*)
|   |   |   |   `-- emath v0.29.1 (*)
|   |   |   |-- emath v0.29.1 (*)
|   |   |   |-- epaint_default_fonts v0.29.1
|   |   |   |-- log v0.4.33
|   |   |   |-- nohash-hasher v0.2.0
|   |   |   `-- parking_lot v0.12.5
|   |   |       |-- lock_api v0.4.14
|   |   |       |   `-- scopeguard v1.2.0
|   |   |       `-- parking_lot_core v0.9.12
|   |   |           |-- cfg-if v1.0.4
|   |   |           |-- smallvec v1.15.2
|   |   |           `-- windows-link v0.2.1
|   |   |-- log v0.4.33
|   |   `-- nohash-hasher v0.2.0
|   |-- egui-winit v0.29.1
|   |   |-- accesskit_winit v0.22.4
|   |   |   |-- accesskit v0.16.3
|   |   |   |-- accesskit_windows v0.23.2
|   |   |   |   |-- accesskit v0.16.3
|   |   |   |   |-- accesskit_consumer v0.24.3
|   |   |   |   |   |-- accesskit v0.16.3
|   |   |   |   |   `-- immutable-chunkmap v2.1.3
|   |   |   |   |       `-- arrayvec v0.7.8
```

```
|   |   |   |   |-- paste v1.0.15 (proc-macro)
|   |   |   |   |-- static_assertions v1.1.0
|   |   |   |   |-- windows v0.58.0
|   |   |   |   |   |-- windows-core v0.58.0
|   |   |   |   |   |   |-- windows-implement v0.58.0 (proc-macro)
|   |   |   |   |   |   |   |-- proc-macro2 v1.0.107 (*)
|   |   |   |   |   |   |   |-- quote v1.0.47 (*)
|   |   |   |   |   |   |   `-- syn v2.0.119 (*)
|   |   |   |   |   |   |-- windows-interface v0.58.0 (proc-macro)
|   |   |   |   |   |   |   |-- proc-macro2 v1.0.107 (*)
|   |   |   |   |   |   |   |-- quote v1.0.47 (*)
|   |   |   |   |   |   |   `-- syn v2.0.119 (*)
|   |   |   |   |   |   |-- windows-result v0.2.0
|   |   |   |   |   |   |   `-- windows-targets v0.52.6
|   |   |   |   |   |   |       `-- windows_x86_64_msvc v0.52.6
|   |   |   |   |   |   |-- windows-strings v0.1.0
|   |   |   |   |   |   |   |-- windows-result v0.2.0 (*)
|   |   |   |   |   |   |   `-- windows-targets v0.52.6 (*)
|   |   |   |   |   |   `-- windows-targets v0.52.6 (*)
|   |   |   |   |   `-- windows-targets v0.52.6 (*)
|   |   |   |   `-- windows-core v0.58.0 (*)
|   |   |   |-- raw-window-handle v0.6.2
|   |   |   `-- winit v0.30.13
|   |   |       |-- bitflags v2.13.1
|   |   |       |-- cursor-icon v1.2.0
|   |   |       |-- dpi v0.1.2
|   |   |       |-- raw-window-handle v0.6.2
|   |   |       |-- smol_str v0.2.2
|   |   |       |-- tracing v0.1.44
|   |   |       |   |-- pin-project-lite v0.2.17
|   |   |       |   `-- tracing-core v0.1.36
|   |   |       |-- unicode-segmentation v1.13.3
|   |   |       `-- windows-sys v0.52.0
|   |   |           `-- windows-targets v0.52.6 (*)
|   |   |       [build-dependencies]
|   |   |       `-- cfg_aliases v0.2.2
|   |   |-- ahash v0.8.12 (*)
|   |   |-- arboard v3.6.1
|   |   |   |-- clipboard-win v5.4.1
|   |   |   |   `-- error-code v3.3.2
|   |   |   |-- log v0.4.33
|   |   |   `-- windows-sys v0.60.2
|   |   |       `-- windows-targets v0.53.5
|   |   |           `-- windows_x86_64_msvc v0.53.1
|   |   |-- egui v0.29.1 (*)
|   |   |-- log v0.4.33
|   |   |-- raw-window-handle v0.6.2
|   |   |-- web-time v1.1.0
|   |   |-- webbrowser v1.2.4
|   |   |   |-- log v0.4.33
|   |   |   `-- url v2.5.8
|   |   |       |-- form_urlencoded v1.2.2
|   |   |       |   `-- percent-encoding v2.3.2
|   |   |       |-- idna v1.1.0
|   |   |       |   |-- idna_adapter v1.2.2
|   |   |       |   |   |-- icu_normalizer v2.2.0
|   |   |       |   |   |   |-- icu_collections v2.2.0
|   |   |       |   |   |   |   |-- displaydoc v0.2.7 (proc-macro)
|   |   |       |   |   |   |   |   |-- proc-macro2 v1.0.107 (*)
|   |   |       |   |   |   |   |   |-- quote v1.0.47 (*)
|   |   |       |   |   |   |   |   `-- syn v3.0.3 (*)
|   |   |       |   |   |   |   |-- potential_utf v0.1.5
|   |   |       |   |   |   |   |   `-- zerovec v0.11.6
|   |   |       |   |   |   |   |       |-- yoke v0.8.3
|   |   |       |   |   |   |   |       |   |-- stable_deref_trait v1.2.1
|   |   |       |   |   |   |   |       |   |-- yoke-derive v0.8.2 (proc-macro)
|   |   |       |   |   |   |   |       |   |   |-- proc-macro2 v1.0.107 (*)
|   |   |       |   |   |   |   |       |   |   |-- quote v1.0.47 (*)
|   |   |       |   |   |   |   |       |   |   |-- syn v2.0.119 (*)
|   |   |       |   |   |   |   |       |   |   `-- synstructure v0.13.2
|   |   |       |   |   |   |   |       |   |       |-- proc-macro2 v1.0.107 (*)
```

```
|   |   |       |   |   |   |   |       |   |       |-- quote v1.0.47 (*)
|   |   |       |   |   |   |   |       |   |       `-- syn v2.0.119 (*)
|   |   |       |   |   |   |   |       |   `-- zerofrom v0.1.8
|   |   |       |   |   |   |   |       |       `-- zerofrom-derive v0.1.7 (proc-macro)
|   |   |       |   |   |   |   |       |           |-- proc-macro2 v1.0.107 (*)
|   |   |       |   |   |   |   |       |           |-- quote v1.0.47 (*)
|   |   |       |   |   |   |   |       |           |-- syn v2.0.119 (*)
|   |   |       |   |   |   |   |       |           `-- synstructure v0.13.2 (*)
|   |   |       |   |   |   |   |       |-- zerofrom v0.1.8 (*)
|   |   |       |   |   |   |   |       `-- zerovec-derive v0.11.3 (proc-macro)
|   |   |       |   |   |   |   |           |-- proc-macro2 v1.0.107 (*)
|   |   |       |   |   |   |   |           |-- quote v1.0.47 (*)
|   |   |       |   |   |   |   |           `-- syn v2.0.119 (*)
|   |   |       |   |   |   |   |-- utf8_iter v1.0.4
|   |   |       |   |   |   |   |-- yoke v0.8.3 (*)
|   |   |       |   |   |   |   |-- zerofrom v0.1.8 (*)
|   |   |       |   |   |   |   `-- zerovec v0.11.6 (*)
|   |   |       |   |   |   |-- icu_normalizer_data v2.2.0
|   |   |       |   |   |   |-- icu_provider v2.2.0
|   |   |       |   |   |   |   |-- displaydoc v0.2.7 (proc-macro) (*)
|   |   |       |   |   |   |   |-- icu_locale_core v2.2.0
|   |   |       |   |   |   |   |   |-- displaydoc v0.2.7 (proc-macro) (*)
|   |   |       |   |   |   |   |   |-- litemap v0.8.2
|   |   |       |   |   |   |   |   |-- tinystr v0.8.3
|   |   |       |   |   |   |   |   |   |-- displaydoc v0.2.7 (proc-macro) (*)
|   |   |       |   |   |   |   |   |   `-- zerovec v0.11.6 (*)
|   |   |       |   |   |   |   |   |-- writeable v0.6.3
|   |   |       |   |   |   |   |   `-- zerovec v0.11.6 (*)
|   |   |       |   |   |   |   |-- writeable v0.6.3
|   |   |       |   |   |   |   |-- yoke v0.8.3 (*)
|   |   |       |   |   |   |   |-- zerofrom v0.1.8 (*)
|   |   |       |   |   |   |   |-- zerotrie v0.2.4
|   |   |       |   |   |   |   |   |-- displaydoc v0.2.7 (proc-macro) (*)
|   |   |       |   |   |   |   |   |-- yoke v0.8.3 (*)
|   |   |       |   |   |   |   |   `-- zerofrom v0.1.8 (*)
|   |   |       |   |   |   |   `-- zerovec v0.11.6 (*)
|   |   |       |   |   |   |-- smallvec v1.15.2
|   |   |       |   |   |   `-- zerovec v0.11.6 (*)
|   |   |       |   |   `-- icu_properties v2.2.0
|   |   |       |   |       |-- icu_collections v2.2.0 (*)
|   |   |       |   |       |-- icu_locale_core v2.2.0 (*)
|   |   |       |   |       |-- icu_properties_data v2.2.0
|   |   |       |   |       |-- icu_provider v2.2.0 (*)
|   |   |       |   |       |-- zerotrie v0.2.4 (*)
|   |   |       |   |       `-- zerovec v0.11.6 (*)
|   |   |       |   |-- smallvec v1.15.2
|   |   |       |   `-- utf8_iter v1.0.4
|   |   |       `-- percent-encoding v2.3.2
|   |   `-- winit v0.30.13 (*)
|   |-- egui_glow v0.29.1
|   |   |-- ahash v0.8.12 (*)
|   |   |-- bytemuck v1.25.2 (*)
|   |   |-- egui v0.29.1 (*)
|   |   |-- glow v0.14.2
|   |   |-- log v0.4.33
|   |   `-- memoffset v0.9.1
|   |       [build-dependencies]
|   |       `-- autocfg v1.5.1
|   |-- glow v0.14.2
|   |-- glutin v0.32.3
|   |   |-- bitflags v2.13.1
|   |   |-- glutin_egl_sys v0.7.1
|   |   |   `-- windows-sys v0.52.0 (*)
|   |   |   [build-dependencies]
|   |   |   `-- gl_generator v0.14.0
|   |   |       |-- khronos_api v3.1.0
|   |   |       |-- log v0.4.33
|   |   |       `-- xml-rs v0.8.29
|   |   |-- glutin_wgl_sys v0.6.1
|   |   |   [build-dependencies]
|   |   |   `-- gl_generator v0.14.0 (*)
```

```
|   |   |-- libloading v0.8.9
|   |   |   `-- windows-link v0.2.1
|   |   |-- once_cell v1.21.4
|   |   |-- raw-window-handle v0.6.2
|   |   `-- windows-sys v0.52.0 (*)
|   |   [build-dependencies]
|   |   `-- cfg_aliases v0.2.2
|   |-- glutin-winit v0.5.0
|   |   |-- glutin v0.32.3 (*)
|   |   |-- raw-window-handle v0.6.2
|   |   `-- winit v0.30.13 (*)
|   |   [build-dependencies]
|   |   `-- cfg_aliases v0.2.2
|   |-- image v0.25.10
|   |   |-- bytemuck v1.25.2 (*)
|   |   |-- byteorder-lite v0.1.0
|   |   |-- moxcms v0.8.1
|   |   |   |-- num-traits v0.2.19 (*)
|   |   |   `-- pxfm v0.1.30
|   |   |-- num-traits v0.2.19 (*)
|   |   `-- png v0.18.1
|   |       |-- bitflags v2.13.1
|   |       |-- crc32fast v1.5.0 (*)
|   |       |-- fdeflate v0.3.7
|   |       |   `-- simd-adler32 v0.3.10
|   |       |-- flate2 v1.1.9
|   |       |   |-- crc32fast v1.5.0 (*)
|   |       |   `-- miniz_oxide v0.8.9
|   |       |       |-- adler2 v2.0.1
|   |       |       `-- simd-adler32 v0.3.10
|   |       `-- miniz_oxide v0.8.9 (*)
|   |-- log v0.4.33
|   |-- parking_lot v0.12.5 (*)
|   |-- raw-window-handle v0.6.2
|   |-- static_assertions v1.1.0
|   |-- web-time v1.1.0
|   |-- winapi v0.3.9
|   |-- windows-sys v0.52.0 (*)
|   `-- winit v0.30.13 (*)
|-- egui_plot v0.29.0
|   |-- ahash v0.8.12 (*)
|   |-- egui v0.29.1 (*)
|   `-- emath v0.29.1 (*)
|-- flate2 v1.1.9 (*)
|-- rand v0.8.7
|   |-- rand_chacha v0.3.1
|   |   |-- ppv-lite86 v0.2.21
|   |   |   `-- zerocopy v0.8.56 (*)
|   |   `-- rand_core v0.6.4
|   |       `-- getrandom v0.2.17
|   |           `-- cfg-if v1.0.4
|   `-- rand_core v0.6.4 (*)
|-- rfd v0.15.4
|   |-- log v0.4.33
|   |-- raw-window-handle v0.6.2
|   `-- windows-sys v0.59.0
|       `-- windows-targets v0.52.6 (*)
|-- serde v1.0.229
|   |-- serde_core v1.0.229
|   `-- serde_derive v1.0.229 (proc-macro)
|       |-- proc-macro2 v1.0.107 (*)
|       |-- quote v1.0.47 (*)
|       `-- syn v3.0.3 (*)
|-- serde_json v1.0.151
|   |-- itoa v1.0.18
|   |-- memchr v2.8.3
|   |-- serde_core v1.0.229
|   `-- zmij v1.0.23
|-- serialport v4.9.0
|   |-- cfg-if v1.0.4
|   |-- scopeguard v1.2.0
```

```
|   `-- windows-sys v0.52.0 (*)
|-- sha2 v0.10.9
|   |-- cfg-if v1.0.4
|   |-- cpufeatures v0.2.17
|   `-- digest v0.10.7
|       |-- block-buffer v0.10.4
|       |   `-- generic-array v0.14.7
|       |       `-- typenum v1.20.1
|       |       [build-dependencies]
|       |       `-- version_check v0.9.5
|       `-- crypto-common v0.1.7
|           |-- generic-array v0.14.7 (*)
|           `-- typenum v1.20.1
`-- thiserror v2.0.20
    `-- thiserror-impl v2.0.20 (proc-macro)
        |-- proc-macro2 v1.0.107 (*)
        |-- quote v1.0.47 (*)
        `-- syn v3.0.3 (*)
[dev-dependencies]
|-- criterion v0.5.1
|   |-- anes v0.1.6
|   |-- cast v0.3.0
|   |-- ciborium v0.2.2
|   |   |-- ciborium-io v0.2.2
|   |   |-- ciborium-ll v0.2.2
|   |   |   |-- ciborium-io v0.2.2
|   |   |   `-- half v2.7.1
|   |   |       |-- cfg-if v1.0.4
|   |   |       `-- zerocopy v0.8.56 (*)
|   |   `-- serde v1.0.229 (*)
|   |-- clap v4.6.6
|   |   `-- clap_builder v4.6.6
|   |       |-- anstyle v1.0.14
|   |       `-- clap_lex v1.1.0
|   |-- criterion-plot v0.5.0
|   |   |-- cast v0.3.0
|   |   `-- itertools v0.10.5
|   |       `-- either v1.17.0
|   |-- is-terminal v0.4.17
|   |   `-- windows-sys v0.61.2
|   |       `-- windows-link v0.2.1
|   |-- itertools v0.10.5 (*)
|   |-- num-traits v0.2.19 (*)
|   |-- once_cell v1.21.4
|   |-- oorandom v11.1.5
|   |-- plotters v0.3.7
|   |   |-- num-traits v0.2.19 (*)
|   |   |-- plotters-backend v0.3.7
|   |   `-- plotters-svg v0.3.7
|   |       `-- plotters-backend v0.3.7
|   |-- rayon v1.12.0
|   |   |-- either v1.17.0
|   |   `-- rayon-core v1.13.0
|   |       |-- crossbeam-deque v0.8.7
|   |       |   |-- crossbeam-epoch v0.9.20
|   |       |   |   `-- crossbeam-utils v0.8.22
|   |       |   `-- crossbeam-utils v0.8.22
|   |       `-- crossbeam-utils v0.8.22
|   |-- regex v1.13.1
|   |   |-- regex-automata v0.4.18
|   |   |   `-- regex-syntax v0.8.11
|   |   `-- regex-syntax v0.8.11
|   |-- serde v1.0.229 (*)
|   |-- serde_derive v1.0.229 (proc-macro) (*)
|   |-- serde_json v1.0.151 (*)
|   |-- tinytemplate v1.2.1
|   |   |-- serde v1.0.229 (*)
|   |   `-- serde_json v1.0.151 (*)
|   `-- walkdir v2.5.0
|       |-- same-file v1.0.6
|       |   `-- winapi-util v0.1.11
```

```
|       |       `-- windows-sys v0.61.2 (*)
|       `-- winapi-util v0.1.11 (*)
`-- proptest v1.11.0
    |-- bit-set v0.8.0
    |   `-- bit-vec v0.8.0
    |-- bit-vec v0.8.0
    |-- bitflags v2.13.1
    |-- num-traits v0.2.19 (*)
    |-- rand v0.9.5
    |   `-- rand_core v0.9.5
    |       `-- getrandom v0.3.4
    |           `-- cfg-if v1.0.4
    |-- rand_chacha v0.9.0
    |   |-- ppv-lite86 v0.2.21 (*)
    |   `-- rand_core v0.9.5 (*)
    |-- rand_xorshift v0.4.0
    |   `-- rand_core v0.9.5 (*)
    |-- regex-syntax v0.8.11
    |-- rusty-fork v0.3.1
    |   |-- fnv v1.0.7
    |   |-- quick-error v1.2.3
    |   |-- tempfile v3.27.0
    |   |   |-- fastrand v2.5.0
    |   |   |-- getrandom v0.4.3
    |   |   |   `-- cfg-if v1.0.4
    |   |   |-- once_cell v1.21.4
    |   |   `-- windows-sys v0.61.2 (*)
    |   `-- wait-timeout v0.2.1
    |-- tempfile v3.27.0 (*)
    `-- unarray v0.1.4
```

</details>

## Working Tree Status

```
M Cargo.lock
 M Cargo.toml
 M docs/ECM-Data.md
 M src/bin/simulator_cli.rs
 M src/io/hw.rs
 M src/io/live_runner.rs
 M src/io/mod.rs
 M src/io/vendor_cat_comm.rs
 M src/lib.rs
 M src/main.rs
 M tests/io_mock.rs
?? docs/CAT_COMM_TEMPLATE_PROTOCOL.md
?? docs/FULL_BOOK_ULTIMATE.md
?? docs/FULL_BOOK_ULTIMATE.pdf
?? docs/FULL_BOOK_V2_CHANGE_ENGINEERING.md
?? docs/FULL_BOOK_V2_CHANGE_ENGINEERING.pdf
?? docs/FULL_DESIGN_CODE_BOOK.md
?? docs/FULL_DESIGN_CODE_BOOK.pdf
?? docs/OSS_INTEGRATION_MATRIX.md
?? reports/
?? scripts/build_cat_comm_template_bridge.ps1
?? scripts/generate_change_engineering_v2.py
?? scripts/generate_design_code_book.py
?? scripts/generate_full_book_ultimate.py
?? scripts/run_windows_template_e2e.ps1
?? scripts/validate_can_utils_interop.sh
?? src/bin/cat_comm_bridge.rs
?? src/io/artifact.rs
?? src/io/production_program.rs
?? tests/vendor_bridge_e2e.rs
```

## Master File Catalog

<details>

<summary>Expand full file catalog with hashes</summary>

- .github/pull_request_template.md | 586 bytes | sha256=d5d9dc276c2f341ec9797bf8a0dc0de2c37ec9a5d727b13156366bcff14d95c3
- .github/workflows/ci.yml | 496 bytes | sha256=2c633e1d4a0eb3e6d57700700d1a3e02b34a60f25288e25a27bce370d5287093
- .github/workflows/fuzz-smoke.yml | 431 bytes | sha256=ad08e2aa78aa0fee40317ed57fae0323b8f58f4036ec7e60abf0667c6f735d11
- .github/workflows/integration-mock.yml | 1107 bytes | sha256=1d11af08f0ad2665190b498100cd79d9c3b4c1b999e378bbc6617a01e584dc96
- .github/workflows/nightly-montecarlo.yml | 680 bytes | sha256=bbedb61b5cb73cafecbf6576c61a511296433d1ac2d54af72f044ca91c00e812
- AUDIT_FULL_2026-08-12.md | 7407 bytes | sha256=93742eecfe1bddb2dd0f0de5a88705791614277087feb8672d3c1f984a61acae
- CHANGELOG.md | 730 bytes | sha256=4a6de389ef2c0fd0a539078e2611333c16073600778ad1d04977a954e0f71c5d
- CODE_EXAMPLES.md | 14462 bytes | sha256=ccd89dc38dd8d538d7db42753cc444915402accdf23936f9334357da3e1d3421
- Cargo.toml | 978 bytes | sha256=92c631d0ca0dfcfd8ce329359fbaca55b93c026f682e9b8095989514e7a72f8b
- J1939_CAN_MANUAL.md | 57299 bytes | sha256=efa3b80417b6edccf41be6c9e00e5f7000e3adba94d7f79a9b32d3d0333008d5
- PROJECT_SUMMARY.md | 9927 bytes | sha256=01314cf9bc4d41f3da72e9935a60701a788efbbd1567ccf63ad72ff1bb2cc727
- README.md | 8333 bytes | sha256=9e215d14de939f38af8478c0b55775f0f02062345d15f7820aea7dc8d6a95a0b
- TESTING.md | 6437 bytes | sha256=f299cb5427211398d3a252304b99e10a9e40caa5b1415afc64caf67dd739edee
- TUTORIALS.md | 17225 bytes | sha256=b956c49a4d9da4362a7283090bd1be93b3d4c934d4f730baf63a3cffa086bd3c
- allowlist.example.json | 358 bytes | sha256=05809313f89ceb70a2a7c756bcd67008db3f29d2bc9404c38620aa4099fe7bd3
- benches/solver_bench.rs | 929 bytes | sha256=a6b09f289be679fbac7a39363fad54e062958090e148588de49d0a08796ab527
- docs/AI_AGENT_RUNBOOK.md | 1039 bytes | sha256=9a917a00c5f8a9b5a56ff92bc71cc76c570150da02393e1828a6857c7e1efb73
- docs/CAT_COMM_TEMPLATE_PROTOCOL.md | 3798 bytes | sha256=0b9cbd017c6c64c9b244d7702bb46afa779899d484ebb8478f8e333409e9d30b
- docs/DECISIONS.md | 771 bytes | sha256=91f5f16ada95796ffd062ba2e44b4b55184a433106275986b9a27d27f4ef0b93
- docs/ECM-Data.md | 7978 bytes | sha256=c28c4591860455f08b998bb144cd69ed784dc92def05699e88a71d293f05be8a
- docs/FULL_BOOK_ULTIMATE.md | 4548691 bytes | sha256=b77513fe39d857ad4feaebf77ec60a2a35b7cffeedd79bc475599b1fde34a3e4

- docs/FULL_BOOK_V2_CHANGE_ENGINEERING.md | 63848 bytes | sha256=23f8914738a29c2ea70cee822c1b8f8bbafd616e677040cee27f23c05ab1267e
- docs/FULL_DESIGN_CODE_BOOK.md | 4598150 bytes | sha256=47f9d203a0ad1764876138562284183 63b3493fc10988cd225da935766760b3a
- docs/OSS_INTEGRATION_MATRIX.md | 7667 bytes | sha256=6dcada9dee5e36090aaaffaca6a9e92230eeaf19e7b213bbbfad1c79b5006efb
- fuzz/Cargo.toml | 340 bytes | sha256=eb7881c64f13c5d5ae34995b912d28ac5f60a27b93bab274bc3f7f2bf0c6195c
- fuzz/fuzz_targets/j1939_frame_from_raw.rs | 453 bytes | sha256=de307820884b7d5bfd3829700af4b1ee591e6eb0c138565e3074b34df8c88ded
- reports/leak_predictions_20260812_095534.json | 107702 bytes | sha256=6fee305dc6955b1ebd38e355181a5419e1f82663162f4725f3ea131b247a2d8c
- reports/leak_report_20260812_095535.json | 1387 bytes | sha256=92aad43ae033a6686fbe2e5a9182041b2e526bb006bfcad20ae43e266eb23fda
- reports/production_phase_report_1786621709828.json | 1985 bytes | sha256=48944c87e197f28bfeede7bb76693ad7a2bd4f2e0b504003ea5a227b0db54b50
- reports/production_phase_report_1786621709828.md | 1044 bytes | sha256=9b36999852e930962dc0b1d47a589af05b615e69c14871bd88582074db9a5e43
- reports/production_phase_report_1786621988419.json | 2447 bytes | sha256=24f3aa01a83044206c77b260559e5521864d88b9bd3583d43092c2d98e84fb5d
- reports/production_phase_report_1786621988419.md | 1217 bytes | sha256=1ac279a04649309b431e66c630d715d293123fd7d81d272bad2be23ba6c194c4
- reports/production_phase_report_1786622089893.json | 2447 bytes | sha256=e7266bb28ef82cc4f9be7c7c1851428d66a2847708d3947a056365f33c9f3c46
- reports/production_phase_report_1786622089893.md | 1217 bytes | sha256=ebf6ac849a68240a72909c75a781e6c34eb42c3b7d59cfe5451e7dcd5e8dd430
- reports/production_phase_report_1786622112533.json | 2623 bytes | sha256=a243750b5adfcb70b1f5396f2a4f9a84eb2b74613513da53d8768e3810a6659f
- reports/production_phase_report_1786622112533.md | 1208 bytes | sha256=63a7fdb13af6cc432468a51fa31e2b8a301540d3624285e0d049fa1183f795a2
- reports/production_phase_report_1786622154581.json | 2627 bytes | sha256=9e9f1cf6896f6ef8be1ad7a7bb925ac182f9aa38aca3ea2732ed45e292d89dc0
- reports/production_phase_report_1786622154581.md | 1212 bytes | sha256=c8ec6c33c4377235cf1e323a81187c7a30768659ceddbc962faec945356e2bd1
- reports/production_phase_report_1786622968863.json | 2995 bytes | sha256=1c15213760e28e9b92e05ebbf4f7efc7aa5ab89750c1830617f78747e215b3eb
- reports/production_phase_report_1786622968863.md | 1461 bytes | sha256=adfb4b3f9e0c1868b97f293908448f8bfde60b98be5357c89ef7bcc03171f7fb
- reports/production_phase_report_1786623531098.json | 3890 bytes | sha256=494296b3f287f9f73db3f76d70eccb153bdffa4287fb516a74fba3054c4e0fb9
- reports/production_phase_report_1786623531098.md | 1646 bytes | sha256=3f4154965f5d2d2bd9644e7da427019e4df7331d91f13a192e19d52cfc9febc2

- scripts/build_cat_comm_template_bridge.ps1 | 946 bytes | sha256=8f8641571fe6f84c268f64cb2e60dc392003dd63de560edb4643d21761f713aa
- scripts/replay_from_artifact.sh | 235 bytes | sha256=5e00cbd45d29e679c0bc75231197897e5781f9b8a47c7d64c9e316d94cadbc98
- scripts/run_fuzz_smoke.sh | 205 bytes | sha256=744d407e851095311b70688cb5ba1ee2d68b1d776ca1e621b0c7607a8fd0ccff
- scripts/run_integration.sh | 81 bytes | sha256=edfca176cb0022453e48da341077cede0a3478592bad1519148ec2a39b193dbf
- scripts/run_monte_carlo.sh | 436 bytes | sha256=55d645546ae8cac673456690671844a1f36ad2afff7c72832a0f5599be88095b
- scripts/run_windows_template_e2e.ps1 | 2004 bytes | sha256=2e24de0356de20fc6b69cb6f7c911adef2e9d380f94150d0027b310b69847b66
- scripts/validate_can_utils_interop.sh | 1662 bytes | sha256=cb8ab23bc4fcbb37604a940e09c0ce00d8eacd4f6f7f060589804a33d3390df3
- src/adas.rs | 8646 bytes | sha256=8a37329b4d07fa9135755aa8d3b8f31e8688e56b1234eb1a6b3f6e3669efae16
- src/autonomous.rs | 22958 bytes | sha256=684025a85050b43765097efeef5592126f962842cf2ee504564ff679b542d537
- src/bin/cat_comm_bridge.rs | 13071 bytes | sha256=05b03f38189191444d22c490f63ccd09c39bf8e031daa94c56088aa995fcc659
- src/bin/simulator_cli.rs | 4789 bytes | sha256=33b70dc1346bcc4346265d61e28fef1fb36a8d170d70c2b3a8072eb247a2766d
- src/boot_sequence.rs | 18107 bytes | sha256=ec1d594062dab92335e5fc2e4e7d2c339468540e499782aaafc576053712c64d
- src/camera.rs | 21361 bytes | sha256=7bde1e7d9bedaa8b10d991d5e9ac608dd039627dee15a1756542bc79dcfcf933
- src/can_bus.rs | 4639 bytes | sha256=3b3b84108fb8b34e22491a4b8c67026fb7ab97f5963bff531c78a5a103551dd9
- src/can_gateway.rs | 10864 bytes | sha256=03d8872587c287c7494bebf50f0de3967512975fde4e581916bc44c0b2c93297
- src/can_network.rs | 47860 bytes | sha256=7495633c9ef45fa755d6f3a2db3eb319588b2cec1a78487aee28a84274e0658f
- src/chassis.rs | 6392 bytes | sha256=687c36266ef1c09c3eb0252460c89180eef46590fb3864461a32169bbd93abb9
- src/ecm_heavy.rs | 13495 bytes | sha256=8109840920ef8e04b4fc983dc268ad50284c745cdb6e04017b44c87f722085a6
- src/ecm_mock_provider.rs | 6555 bytes | sha256=55474d10d3b21983d856f7d597f461715c9576500be00d1bb75f3199dfe9d4e9
- src/ecu_abs.rs | 20916 bytes | sha256=24e61badc2eefc55d89592712370908d5606ba2b3aeb457dec8e759740841a73
- src/ecu_bcm.rs | 12750 bytes | sha256=c41724dfa5784a5414d1eea9bc21da6e368e594ace06bc4cc0d59591aaf901b1
- src/ecu_ecm.rs | 32737 bytes | sha256=118aaccdba945a23ea75eb0ca87454ede929d08a8165b9448f80d542fd2ddb71

- src/ecu_hcm.rs | 18611 bytes | sha256=e5e597917322c7263a98c34bea645d76813024f200b830896807bc5a1ccb4c31
- src/ecu_icm.rs | 17673 bytes | sha256=529ec9620c52eb7250a211980ea30a9632c1918af1e3b280c27f594d399268f2
- src/ecu_tcm.rs | 30606 bytes | sha256=1aa30892971e890c9c641fb4c65ec9ab49637831df50f9356e6687865b6af50a
- src/ecu_vcm.rs | 16600 bytes | sha256=aa24a0e4ec759096d6c84f65aed733d0d348aefcc01ef5f3e177adfac3cfe871
- src/electrical.rs | 17958 bytes | sha256=bfecba34d809c7ed10799fd6d8d65918c77e3fbf8eff7987101a4eeeab408b67
- src/engine.rs | 4549 bytes | sha256=6360476aacaf59a383d207fb3962f2d3cbfe9775aaeaa46dac34bf36c1fb1282
- src/gps.rs | 16060 bytes | sha256=e204e1bd9b2060e7cce8fca87e0127d69b94eeb1a2fa1d1d569433ecb88d33e1
- src/implement.rs | 20426 bytes | sha256=44b3d1ab7bc9a9b18b787ef5cc5fdad83004a38019c3abce1a6c87c3acfd9521
- src/imu.rs | 14582 bytes | sha256=993e491a430dd2d50a47a9bbf89232f00a568b6096b545f97706609238e07bf3
- src/io/allowlist.rs | 7353 bytes | sha256=942643b0bcfbcd65cd1a3be668c8b55bf7d9eeefe2070b6fe1dc44ff0da03609
- src/io/artifact.rs | 1401 bytes | sha256=a5bfb29fce6224cac25b0859c7ff01b0572b9d82b472733bf134a8aaa50c51d9
- src/io/ecm_params.rs | 5056 bytes | sha256=0f1e040bf3ae5c52adead9050850e080c70cec5b1770e87c6d7d8c4cf021c123
- src/io/hw.rs | 14612 bytes | sha256=93da8a4d8d8648b01976092be492ea34e78b65383713ff91708a87ca44709471
- src/io/live_runner.rs | 53004 bytes | sha256=febbe28208d26806ccc94d6418a0dfd642c1e8222decef4aa3ebf9ae3913d8dc
- src/io/metrics.rs | 1324 bytes | sha256=6f2f6db6fbcd3f2b0cedf8efe3b7e50d6869c1d71e355d2362116f879b660ed8
- src/io/mock.rs | 5443 bytes | sha256=3d7920445ada44c38a08d4b2a370b73e677a43bcd5341c0ae8b94503c573c76f
- src/io/mod.rs | 359 bytes | sha256=98f824a24cb2367af8d1aa5dcc5bd281bc8a223cf894a564ce6c4805c40e9d0f
- src/io/production_program.rs | 28742 bytes | sha256=5b38af4945a9d301831820ee2508d1865614851c68c9a5a77acc684ec272076d
- src/io/rate_limiter.rs | 2113 bytes | sha256=8fb3a1a275bb3216b809fd7258822c433e0bd43ae25a6e00b1b12a4a522ea411
- src/io/replay.rs | 5228 bytes | sha256=567a424db9390ee10fa18afdfd20d3f3018585755aa15c2ed6c5adb27b8b3960
- src/io/serial_adapter.rs | 2834 bytes | sha256=39cf0345f771f51fb8a3608d58b9b2540a4289e33f3c1a5b30022fa02aa27d03
- src/io/socketcan_adapter.rs | 4355 bytes | sha256=c2a6e64dd2272e4943d2898402ca9d2cc0c2ee6a721a58ecc719ae31729ba8be
- src/io/vendor_cat_comm.rs | 13729 bytes | sha256=81a7528092c84c57fff839afeb5e145caf97c3220debef73f83455912cc70d59

- src/j1939.rs | 51087 bytes | sha256=af61a7339bd56639682c377dbb549c4162b4a941147bf85f33e73dd3defbb763
- src/leak_physics.rs | 117808 bytes | sha256=b795d34c11e02aa7fc423490c33553584166c2fa8693bbfb8bf436c2de63c6ba
- src/lib.rs | 47349 bytes | sha256=d12d994466f4696f88fcc3b005ead0cc9156ecf676b21757e8fa02dbd7652c8c
- src/lidar.rs | 13441 bytes | sha256=af8c72ea9ccc4ff3463c8e565e3f699bbac3183e4f4c805c5210535d79a9f5be
- src/main.rs | 439936 bytes | sha256=2915d60c68c099cd701e604c724958f12ec2f8701df5230d563d090c658f8801
- src/network_mgmt.rs | 12675 bytes | sha256=47918f0667cd6becd81a2bf1724396131077a680cbe80fcf846db4d536cabfc8
- src/nvm.rs | 14367 bytes | sha256=be385cd47c40f678ee61d8c9a8fb0384bbfcec03ce0971d5d271fd6aefc46316
- src/observability.rs | 1408 bytes | sha256=e5c97a9c1ad342d20cb9234ffad9458a1683e3f149b1550db635a05a621dff9a
- src/radar.rs | 13581 bytes | sha256=ef696891c0981022b1696cc4e47fdb83048f8a68391e442fdfcec8543d149a6c
- src/sensors.rs | 5918 bytes | sha256=0a1c9b908519ce5f80e2521f95adfc384537dfc85d79e8cd0aafb35e5483c5b9
- src/sim_core.rs | 1987 bytes | sha256=b50383ed60e704f1ef1643c7d30c8c157d149d7b23b8294ecf2f870ed5f34ae5
- src/transmission.rs | 4076 bytes | sha256=2cf8fa43485654936351dc8b15ba2036d831ad6523d291d02ecfd86a351ee3ea
- src/uds.rs | 40379 bytes | sha256=c4289959f87ecbb53872060ec73ed03bbfdbbebc4ec325030fab71f24cbf1eb0
- src/v2x_telematics.rs | 23423 bytes | sha256=43bcbceb658a3e70e6cc37a4892643b57ab9a07863f54ff65a3167592a95abe9
- src/vehicle.rs | 4416 bytes | sha256=ae852613e024acaaa586ccfb5073809ce8eab080905470af29422870d33ecb30
- src/widgets.rs | 10605 bytes | sha256=eb9832ae5258f317e70b9b56e46d5f31811ec9c8e4c9bfa84b74725619a944b6
- tests/io_mock.rs | 6416 bytes | sha256=843958f0bdde380799f789abbdd4ace026d5d991bf769a9065afb33e7d5bcd6e
- tests/leak_system_integration.rs | 2947 bytes | sha256=972d0ba25fe7c7c66b0185c2fcf6909a88e1a21fbe61b7b9f4c57cf2b3cdc4fa
- tests/property_invariants.rs | 1276 bytes | sha256=f985a14ec46ce3484d7cfa9ae8cb569fbccd69052ee936ab96e64c29fc741916
- tests/speed_regression.rs | 1579 bytes | sha256=ad71f6dcf618c173d9d21a57f51ce6fd99df701214934ac69b59dad761eabbd5
- tests/system_failure_scenarios.rs | 2094 bytes | sha256=8ef79e52934433df827b035ed81f451511e01f768043c4ac16599cd4798812a2
- tests/vendor_bridge_e2e.rs | 3195 bytes | sha256=0c368c59c8a987e249143d1676d6835b2b39218a963fa0e56781b33e30af3dc2

</details>

## File Deep Dive: .github/pull_request_template.md

Role: CI/CD governance and continuous validation

Line count: 27 | Non-empty: 23 | Symbol count: 5

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h2 | Problem | namespace and compile-time linkage |
| 4 | h2 | Changes | namespace and compile-time linkage |
| 10 | h2 | How to test | namespace and compile-time linkage |
| 17 | h2 | Artifacts | namespace and compile-time linkage |
| 21 | h2 | Acceptance checklist | namespace and compile-time linkage |
```

### Content Snapshot

```
## Problem
Describe the issue and impact.

## Changes
- [ ] Refactor/API-preserve
- [ ] Tests
- [ ] Docs/Changelog
- [ ] Observability

## How to test
```

cargo fmt -- --check

cargo clippy -- -D warnings

cargo test

```
## Artifacts
- Bench logs:
- Replay/Monte Carlo:

## Acceptance checklist
- [ ] Changelog entry included
- [ ] Unit tests for changed logic
- [ ] Integration tests where applicable
- [ ] No public API breakage (or compatibility shim included)
- [ ] Performance impact documented
- [ ] Rollback path documented (feature flag or revert)
```

## File Deep Dive: .github/workflows/ci.yml

Role: CI/CD governance and continuous validation

Line count: 21 | Non-empty: 19 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | name: CI | support logic; impact depends on neighboring block and data flow. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | on: | support logic; impact depends on neighboring block and data flow. |
| 4 |   push: | support logic; impact depends on neighboring block and data flow. |
| 5 |   pull_request: | support logic; impact depends on neighboring block and data flow. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | jobs: | support logic; impact depends on neighboring block and data flow. |
| 8 |   lint-and-unit: | support logic; impact depends on neighboring block and data flow. |
| 9 |     runs-on: ubuntu-latest | support logic; impact depends on neighboring block and data flow. |
| 10 |     steps: | support logic; impact depends on neighboring block and data flow. |
| 11 |       - uses: actions/checkout@v4 | support logic; impact depends on neighboring block and data flow. |
| 12 |       - name: Setup Rust | support logic; impact depends on neighboring block and data flow. |
| 13 |         uses: dtolnay/rust-toolchain@stable | support logic; impact depends on neighboring block and data flow.
| 14 |         with: | support logic; impact depends on neighboring block and data flow. |
```

```
| 15 |           components: rustfmt, clippy | support logic; impact depends on neighboring block and data flow. |
| 16 |       - name: Format check | support logic; impact depends on neighboring block and data flow. |
| 17 |         run: cargo fmt -- --check | support logic; impact depends on neighboring block and data flow. |
| 18 |       - name: Clippy | support logic; impact depends on neighboring block and data flow. |
| 19 |         run: cargo clippy --workspace --all-targets | support logic; impact depends on neighboring block and data
| 20 |       - name: Unit and integration tests | support logic; impact depends on neighboring block and data flow. |
| 21 |         run: cargo test --locked --workspace | support logic; impact depends on neighboring block and data flow.
```

# File Deep Dive: .github/workflows/fuzz-smoke.yml

Role: CI/CD governance and continuous validation

Line count: 18 | Non-empty: 16 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | name: Fuzz Smoke | support logic; impact depends on neighboring block and data flow. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | on: | support logic; impact depends on neighboring block and data flow. |
| 4 |   schedule: | support logic; impact depends on neighboring block and data flow. |
| 5 |     - cron: "0 5 * * 0" | support logic; impact depends on neighboring block and data flow. |
| 6 |   workflow_dispatch: | support logic; impact depends on neighboring block and data flow. |
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | jobs: | support logic; impact depends on neighboring block and data flow. |
| 9 |   fuzz-smoke: | support logic; impact depends on neighboring block and data flow. |
| 10 |     runs-on: ubuntu-latest | support logic; impact depends on neighboring block and data flow. |
| 11 |     steps: | support logic; impact depends on neighboring block and data flow. |
| 12 |       - uses: actions/checkout@v4 | support logic; impact depends on neighboring block and data flow. |
| 13 |       - name: Setup Rust | support logic; impact depends on neighboring block and data flow. |
| 14 |         uses: dtolnay/rust-toolchain@stable | support logic; impact depends on neighboring block and data flow.
| 15 |       - name: Install cargo-fuzz | support logic; impact depends on neighboring block and data flow. |
| 16 |         run: cargo install cargo-fuzz | support logic; impact depends on neighboring block and data flow. |
| 17 |       - name: Run fuzz smoke | support logic; impact depends on neighboring block and data flow. |
| 18 |         run: cargo fuzz run j1939_frame_from_raw -- -max_total_time=120 | support logic; impact depends on neighb
```

# File Deep Dive: .github/workflows/integration-mock.yml

Role: CI/CD governance and continuous validation

Line count: 40 | Non-empty: 37 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | name: Integration Mock | support logic; impact depends on neighboring block and data flow. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | on: | support logic; impact depends on neighboring block and data flow. |
| 4 |   push: | support logic; impact depends on neighboring block and data flow. |
| 5 |   pull_request: | support logic; impact depends on neighboring block and data flow. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | jobs: | support logic; impact depends on neighboring block and data flow. |
| 8 |   integration-mock: | support logic; impact depends on neighboring block and data flow. |
| 9 |     runs-on: ubuntu-latest | support logic; impact depends on neighboring block and data flow. |
| 10 |     steps: | support logic; impact depends on neighboring block and data flow. |
| 11 |       - uses: actions/checkout@v4 | support logic; impact depends on neighboring block and data flow. |
| 12 |       - name: Setup Rust | support logic; impact depends on neighboring block and data flow. |
| 13 |         uses: dtolnay/rust-toolchain@stable | support logic; impact depends on neighboring block and data flow.
| 14 |         with: | support logic; impact depends on neighboring block and data flow. |
```

```
| 15 |           components: rustfmt, clippy | support logic; impact depends on neighboring block and data flow. |
| 16 |       - name: Format check | support logic; impact depends on neighboring block and data flow. |
| 17 |         run: cargo fmt -- --check | support logic; impact depends on neighboring block and data flow. |
| 18 |       - name: Clippy | support logic; impact depends on neighboring block and data flow. |
| 19 |         run: cargo clippy --test io_mock | support logic; impact depends on neighboring block and data flow. |
| 20 |       - name: Run mock integration tests | support logic; impact depends on neighboring block and data flow. |
| 21 |         run: cargo test --test io_mock | support logic; impact depends on neighboring block and data flow. |
| 22 |       - name: Cargo check | support logic; impact depends on neighboring block and data flow. |
| 23 |         run: cargo check | support logic; impact depends on neighboring block and data flow. |
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 |   integration-mock-windows: | support logic; impact depends on neighboring block and data flow. |
| 26 |     runs-on: windows-latest | support logic; impact depends on neighboring block and data flow. |
| 27 |     steps: | support logic; impact depends on neighboring block and data flow. |
| 28 |       - uses: actions/checkout@v4 | support logic; impact depends on neighboring block and data flow. |
| 29 |       - name: Setup Rust | support logic; impact depends on neighboring block and data flow. |
| 30 |         uses: dtolnay/rust-toolchain@stable | support logic; impact depends on neighboring block and data flow.
| 31 |         with: | support logic; impact depends on neighboring block and data flow. |
| 32 |           components: rustfmt, clippy | support logic; impact depends on neighboring block and data flow. |
| 33 |       - name: Format check | support logic; impact depends on neighboring block and data flow. |
| 34 |         run: cargo fmt -- --check | support logic; impact depends on neighboring block and data flow. |
| 35 |       - name: Clippy (io_mock) | support logic; impact depends on neighboring block and data flow. |
| 36 |         run: cargo clippy --test io_mock | support logic; impact depends on neighboring block and data flow. |
| 37 |       - name: Run mock integration tests | support logic; impact depends on neighboring block and data flow. |
| 38 |         run: cargo test --test io_mock | support logic; impact depends on neighboring block and data flow. |
| 39 |       - name: Cargo check with vendor-windows feature | support logic; impact depends on neighboring block and da
| 40 |         run: cargo check --features vendor-windows | support logic; impact depends on neighboring block and data
```

# File Deep Dive: .github/workflows/nightly-montecarlo.yml

Role: CI/CD governance and continuous validation

Line count: 25 | Non-empty: 23 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | name: Nightly Monte Carlo | support logic; impact depends on neighboring block and data flow. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | on: | support logic; impact depends on neighboring block and data flow. |
| 4 |   schedule: | support logic; impact depends on neighboring block and data flow. |
| 5 |     - cron: "0 3 * * *" | support logic; impact depends on neighboring block and data flow. |
| 6 |   workflow_dispatch: | support logic; impact depends on neighboring block and data flow. |
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | jobs: | support logic; impact depends on neighboring block and data flow. |
| 9 |   nightly-montecarlo: | support logic; impact depends on neighboring block and data flow. |
| 10 |     runs-on: ubuntu-latest | support logic; impact depends on neighboring block and data flow. |
| 11 |     strategy: | support logic; impact depends on neighboring block and data flow. |
| 12 |       matrix: | support logic; impact depends on neighboring block and data flow. |
| 13 |         model: [reduced, high-fidelity] | support logic; impact depends on neighboring block and data flow. |
| 14 |         seed_batch: [1, 2, 3] | support logic; impact depends on neighboring block and data flow. |
| 15 |     steps: | support logic; impact depends on neighboring block and data flow. |
| 16 |       - uses: actions/checkout@v4 | support logic; impact depends on neighboring block and data flow. |
| 17 |       - name: Setup Rust | support logic; impact depends on neighboring block and data flow. |
| 18 |         uses: dtolnay/rust-toolchain@stable | support logic; impact depends on neighboring block and data flow.
| 19 |       - name: Run Monte Carlo script | support logic; impact depends on neighboring block and data flow. |
| 20 |         run: bash scripts/run_monte_carlo.sh 1 ${{ matrix.model }} | support logic; impact depends on neighboring
| 21 |       - name: Upload artifacts | support logic; impact depends on neighboring block and data flow. |
| 22 |         uses: actions/upload-artifact@v4 | support logic; impact depends on neighboring block and data flow. |
| 23 |         with: | support logic; impact depends on neighboring block and data flow. |
| 24 |           name: monte-artifacts-${{ matrix.model }}-${{ matrix.seed_batch }} | support logic; impact depends on
| 25 |           path: artifacts/monte | support logic; impact depends on neighboring block and data flow. |
```

# File Deep Dive: AUDIT_FULL_2026-08-12.md

Role: project support artifact

Line count: 186 | Non-empty: 132 | Symbol count: 12

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---|---|---|---|
| 1 | h1 | Full Technical Audit - AutoBreaking | namespace and compile-time linkage |
| 7 | h2 | 1. What was executed | namespace and compile-time linkage |
| 19 | h2 | 2. Execution evidence summary | namespace and compile-time linkage |
| 38 | h2 | 3. Severity summary | namespace and compile-time linkage |
| 51 | h2 | 4. Confirmed findings (highest priority) | namespace and compile-time linkage |
| 53 | h3 | High | namespace and compile-time linkage |
| 69 | h3 | Medium | namespace and compile-time linkage |
| 103 | h2 | 5. Agent-flagged findings requiring targeted confirmation | namespace and compile-time linkage |
| 112 | h2 | 6. Module-by-module status | namespace and compile-time linkage |
| 156 | h2 | 7. API and integration audit conclusions | namespace and compile-time linkage |
| 162 | h2 | 8. Recommended remediation roadmap | namespace and compile-time linkage |
| 182 | h2 | 9. Final audit verdict | namespace and compile-time linkage |

## Content Snapshot

```
# Full Technical Audit - AutoBreaking

Date: 2026-08-12
Scope: full codebase audit with static review + test/lint/build + deterministic runtime scenarios
Mode: read-only audit with evidence collection and prioritized remediation plan

## 1. What was executed

- `cargo check --workspace`
- `cargo clippy --workspace --all-targets`
- `cargo test --workspace`
- `cargo run --bin simulator_cli -- --seed {1,7,42,99,123456} --steps 12000`
- `cargo run --bin simulator_cli -- --replay sample.can --verify`
- 3 parallel deep audits via Explore agent:
  - Core logic and physics modules
  - Integration/protocol/API modules
  - UI/GUI/UX modules

## 2. Execution evidence summary

- Build: PASS
- Tests: PASS (all workspace tests green)
- Clippy: PASS with warnings only (no build-blocking defects)
- CLI deterministic scenarios: PASS across all seeds tested

Scenario snapshots (12000 steps each):

- seed=1: can_errors=0, can_health=0.9415, fuel_pct=84.1967
- seed=7: can_errors=0, can_health=0.9415, fuel_pct=84.1897
- seed=42: can_errors=0, can_health=0.9415, fuel_pct=84.1958
- seed=99: can_errors=0, can_health=0.9415, fuel_pct=84.1894
- seed=123456: can_errors=0, can_health=0.9415, fuel_pct=84.1981

Replay path:

- `replay_file=sample.can verify=true status=ok`

## 3. Severity summary

Legend:

- CONFIRMED: manually re-checked in source by this audit run
- AGENT-FLAGGED: raised by subagent and needs targeted confirmation test before merge

Totals:

- Critical: 0 CONFIRMED, 3 AGENT-FLAGGED
- High: 2 CONFIRMED, 8 AGENT-FLAGGED
- Medium: 5 CONFIRMED, 12 AGENT-FLAGGED
## 4. Confirmed findings (highest priority)
```

### High

1) Handshake acceptance in live ECM connect is broad and can accept unrelated CAN response source as success.

- File: `src/io/live_runner.rs`
- Function: `connect_ecm`
- Risk: false positive in live connection readiness.
- Action: enforce PGN/SA response validation against expected request set.

2) Background live thread errors are only printed to stderr and not propagated to structured health state.

- File: `src/io/live_runner.rs`
- Function: `start_retrieve_data`
- Risk: silent degraded live data with stale snapshot.
- Action: add shared health/error state and surface to UI + observability.

### Medium

3) Autonomous ACC integral can remain accumulated across traffic state transitions.

- File: `src/autonomous.rs`
- Function: `run_acc`
- Risk: transient overshoot when lead target disappears/reappears.
- Action: reset or decay integral when lead is infinite/unreliable.

4) Clippy indicates maintainability debt in UI control flow.

- File: `src/main.rs`
- Warnings: `match_single_binding`, `useless_format`
- Risk: low runtime impact, higher maintenance cost.
- Action: clean up for readability and safer future edits.

5) UI command burst risk on high-frequency sliders (throttle/brake update cadence).

- File: `src/main.rs`
- Risk: extra queue pressure under sustained drag.
- Action: introduce small debounce/rate-limit for command enqueue.

6) CAN connect path validation should be hardened with timeout/retry reason metadata.

- File: `src/io/live_runner.rs`
- Risk: poor diagnosability in field failures.
- Action: classify failures by stage and expose code/reason.

7) UI state synchronization remains complex and fragile in parameter pages.

- File: `src/main.rs`
- Risk: stale values when dirty-state toggles are incomplete.
- Action: centralize sync policy per tab with explicit apply/cancel model.

## 5. Agent-flagged findings requiring targeted confirmation

These were reported by deep subagents and should be validated with focused reproduction before direct fix:

- Potential ESP correction logic mismatch under oversteer/understeer branch behavior.
- Potential AD tab feedback gaps for ACC/LKA action acknowledgments.
- Potential Leak Lab validation gaps for manual/custom parameter invariants.
- Potential performance hot spots in trace/event rendering under heavy data volume.

## 6. Module-by-module status

Core simulation and vehicle dynamics:

- `src/lib.rs`: PASS runtime checks; integration-heavy, medium complexity risk.
- `src/sim_core.rs`: PASS tests.
- `src/engine.rs`: no critical flags in this audit.
- `src/transmission.rs`: no critical flags in this audit.
- `src/ecu_ecm.rs`: stable under tests; monitor calibration realism.
- `src/ecu_tcm.rs`: recently revised; speed regression tests pass.
- `src/ecu_abs.rs`: no confirmed critical defect; add scenario tests for ESP edge cases.
- `src/ecu_vcm.rs`: no critical flags in this run.

```
- `src/chassis.rs`: no confirmed critical defect in this run.
- `src/implement.rs`: integration stable in tests.
- `src/leak_physics.rs`: broad coverage and integration tests pass.

Autonomy and perception:

- `src/autonomous.rs`: functional path active; medium risk in ACC integral handling.
- `src/adas.rs`: no critical flags in this run.
- `src/sensors.rs`: no critical flags in this run.
- `src/radar.rs`: no critical flags in this run.
- `src/camera.rs`: no critical flags in this run.
- `src/lidar.rs`: no critical flags in this run.
- `src/gps.rs`: no critical flags in this run.
- `src/imu.rs`: no critical flags in this run.

Network, protocol, diagnostics, storage:

- `src/j1939.rs`: tests pass.
- `src/can_bus.rs`: no critical flags in this run.
- `src/can_gateway.rs`: no critical flags in this run.
- `src/can_network.rs`: tests pass.
- `src/network_mgmt.rs`: no critical flags in this run.
- `src/uds.rs`: prior hard fix present; tests pass.
- `src/io/*`: HIGH priority hardening in live runner path.
- `src/nvm.rs`: medium hardening opportunity for wear-limit enforcement policy.
- `src/observability.rs`: medium integration opportunity to wire structured health events.

UI/GUI/UX:

- `src/main.rs`: functionally broad and complex; medium maintainability/performance risks.
- `src/widgets.rs`: no critical flags in this run.

## 7. API and integration audit conclusions

- Physical simulation API surface is coherent and test-backed.
- Live adapter path is the highest production risk area.
- UDS and CAN protocol paths are generally robust after recent fixes, but need deeper contract/fault tests for product

## 8. Recommended remediation roadmap

Phase 1 (urgent, 1-2 days):

- Harden `connect_ecm` response validation in `src/io/live_runner.rs`.
- Add live feed health propagation (thread status, last error, stale window).
- Add focused tests for live connect failure modes.

Phase 2 (short, 2-4 days):

- Add ACC integral reset/decay policy tests in `src/autonomous.rs`.
- Add ESP branch behavior scenario tests in `src/ecu_abs.rs`.
- Clean top clippy warnings in `src/main.rs`.

Phase 3 (stabilization, 1 week):

- UI performance pass for heavy lists/trace panels.
- Structured observability wiring across live I/O and protocol failure points.
- Add contract/fault injection test matrix for `src/io/*` and `src/uds.rs`.

## 9. Final audit verdict

- Current state: functionally healthy and test-stable.
- Production risk concentration: live hardware integration path (`src/io/live_runner.rs`) and large-state UI operabili
- Confidence level: high for compile/test stability, medium for field integration robustness without the hardening act
```

## File Deep Dive: CHANGELOG.md

Role: project support artifact

Line count: 16 | Non-empty: 14 | Symbol count: 4

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Changelog | namespace and compile-time linkage |
| 3 | h2 | [Unreleased] | namespace and compile-time linkage |
| 4 | h3 | Added | namespace and compile-time linkage |
| 14 | h3 | Changed | namespace and compile-time linkage |
```

### Content Snapshot

```
# Changelog

## [Unreleased]
### Added
- CI workflows for lint/unit/nightly Monte Carlo/fuzz smoke.
- Observability module with structured JSON events and simulation metrics.
- Incremental solver architecture module (`sim_core`) with integrator traits.
- Property-based tests for physical bounds and leak invariants.
- Monte Carlo, replay, and integration scripts under `scripts/`.
- Fuzz harness scaffold for J1939/CAN frame parser.
- Benchmark harness for simulation steps (`benches/solver_bench.rs`).
- Agent runbook and PR checklist template.

### Changed
- CAN network now supports health scoring, per-bus injections, and snapshot exports.
- J1939 registry expanded with ADAS/fusion/energy PGNs and builders.
```

## File Deep Dive: CODE_EXAMPLES.md

Role: project support artifact

Line count: 551 | Non-empty: 439 | Symbol count: 43

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | ■ Exemplos de Código - Sistema de Freio com ABS | namespace and compile-time linkage |
| 7 | h2 | 1. Exemplo: Modificar Frequência ABS | namespace and compile-time linkage |
| 9 | h3 | Localização | namespace and compile-time linkage |
| 12 | h3 | Código Original | namespace and compile-time linkage |
| 20 | h3 | Modificação: Aumentar para 10 Hz | namespace and compile-time linkage |
| 36 | h3 | Recompilar e Testar | namespace and compile-time linkage |
| 40 | h1 | Selecione Emergency Brake (tecla 2) | namespace and compile-time linkage |
| 41 | h1 | Observe ciclos ABS acontecerem ~25% mais rápido | namespace and compile-time linkage |
| 46 | h2 | 2. Exemplo: Ajustar Limiar de Travamento | namespace and compile-time linkage |
| 48 | h3 | Localização | namespace and compile-time linkage |
| 51 | h3 | Código Original | namespace and compile-time linkage |
| 59 | h3 | Modificação: Mais Sensível (4 km/h) | namespace and compile-time linkage |
| 67 | h3 | Impacto | namespace and compile-time linkage |
| 74 | h2 | 3. Exemplo: Mudar Pressão de Liberação ABS | namespace and compile-time linkage |
| 76 | h3 | Localização | namespace and compile-time linkage |
| 79 | h3 | Código Original | namespace and compile-time linkage |
| 88 | h3 | Modificação: Mais Agressivo | namespace and compile-time linkage |
| 97 | h3 | Modificação: Mais Conservador | namespace and compile-time linkage |
| 108 | h2 | 4. Exemplo: Adicionar Novo Cenário | namespace and compile-time linkage |
| 110 | h3 | Passo 1: Adicionar Variante de Enum | namespace and compile-time linkage |
| 124 | h3 | Passo 2: Implementar Display | namespace and compile-time linkage |
| 138 | h3 | Passo 3: Implementar Lógica | namespace and compile-time linkage |
| 158 | h3 | Passo 4: Adicionar Tecla de Ativação | namespace and compile-time linkage |
| 169 | h3 | Passo 5: Atualizar Controles (comentário) | namespace and compile-time linkage |
| 179 | h2 | 5. Exemplo: Implementar Função de Análise | namespace and compile-time linkage |
| 181 | h3 | Novo Módulo: metrics.rs | namespace and compile-time linkage |
| 234 | h3 | Usar no main.rs | namespace and compile-time linkage |
| 252 | h2 | 6. Exemplo: Exportar Dados para CSV | namespace and compile-time linkage |
| 254 | h3 | Novo Módulo: data_export.rs | namespace and compile-time linkage |
| 303 | h3 | Usar | namespace and compile-time linkage |
| 312 | h2 | 7. Exemplo: Sistema de Configuração Customizado | namespace and compile-time linkage |
| 314 | h3 | Novo tipo: Config | namespace and compile-time linkage |
| 357 | h3 | Arquivo config.json | namespace and compile-time linkage |
| 373 | h2 | 8. Exemplo: Teste Unitário Customizado | namespace and compile-time linkage |
```

```
| 375 | h3 | Adicionar em lib.rs | namespace and compile-time linkage |
| 436 | h2 | 9. Exemplo: Criar Curva de Performance | namespace and compile-time linkage |
| 438 | h3 | Função: Testar em Múltiplas Velocidades | namespace and compile-time linkage |
| 478 | h2 | 10. Exemplo: Integração com Hardware Simulado | namespace and compile-time linkage |
| 480 | h3 | Pseudocódigo: Conexão CAN Bus | namespace and compile-time linkage |
| 521 | h2 | ■ Checklist de Customização | namespace and compile-time linkage |
| 536 | h2 | ■ Referências Rápidas | namespace and compile-time linkage |
| 538 | h3 | Arquivos Principais | namespace and compile-time linkage |
| 543 | h3 | Documentação | namespace and compile-time linkage |
```

### Content Snapshot

```
# ■ Exemplos de Código - Sistema de Freio com ABS

Coleção de snippets de código úteis e exemplos de extensão.

---

## 1. Exemplo: Modificar Frequência ABS

### Localização
[src/lib.rs](src/lib.rs#L130)

### Código Original
```

pub struct ABSController {

// ...

pulse_frequency: f64, // 8.0 Hz

}

```
### Modificação: Aumentar para 10 Hz
```

impl ABSController {

pub fn new() -> Self {

Self {

brake_system: BrakeSystem::new(),

vehicle_velocity: 0.0,

abs_active: false,

current_pressure: 0.0,

abs_cycle: 0.0,

pulse_frequency: 10.0, // ← Alterado para 10 Hz

}

}

}

```
### Recompilar e Testar
```

cargo build

cargo run

# Selecione Emergency Brake (tecla 2)

# Observe ciclos ABS acontecerem ~25% mais rápido

```
---

## 2. Exemplo: Ajustar Limiar de Travamento

### Localização
[src/lib.rs](src/lib.rs#L88)

### Código Original
```

let velocity_diff = vehicle_velocity - self.velocity;

if velocity_diff > 5.0 && self.brake_pressure > 0.3 { // 5 km/h threshold

self.state = WheelState::Skidding;

}

```
### Modificação: Mais Sensível (4 km/h)
```

// Detecta travamento em 4 km/h em vez de 5 km/h

if velocity_diff > 4.0 && self.brake_pressure > 0.3 {

self.state = WheelState::Skidding;

}

```
### Impacto
- **ABS mais cedo**: Ativa ~0.5s mais rápido
- **Mais ciclos**: ~10-15% mais ciclos ABS
- **Parada mais suave**: Menos "pulsação" perceptível

---

## 3. Exemplo: Mudar Pressão de Liberação ABS

### Localização
[src/lib.rs](src/lib.rs#L162)

### Código Original
```

if self.abs_cycle < 0.5 {

self.current_pressure = base_pressure * 0.3; // 30% liberação

} else {

self.current_pressure = base_pressure * 0.9; // 90% aplicação

}

```
### Modificação: Mais Agressivo
```

if self.abs_cycle < 0.5 {

self.current_pressure = base_pressure * 0.15; // 15% (muito agressivo)

} else {

self.current_pressure = base_pressure * 0.95; // 95% (mais pressão)

}

```
### Modificação: Mais Conservador
```

if self.abs_cycle < 0.5 {

self.current_pressure = base_pressure * 0.4; // 40% (menos agressivo)

} else {

self.current_pressure = base_pressure * 0.85; // 85% (menos pressão)

```
}
---

## 4. Exemplo: Adicionar Novo Cenário

### Passo 1: Adicionar Variante de Enum
[src/main.rs](src/main.rs#L103)
```

#[derive(Debug, Clone, Copy, PartialEq)]

enum SimulationScenario {

Manual,

EmergencyBrake,

HighSpeed,

RepeatedBraking,

WetRoad, // ← NOVO

}

```
### Passo 2: Implementar Display
[src/main.rs](src/main.rs#L113)
```

impl std::fmt::Display for SimulationScenario {

fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result {

match self {

// ... casos existentes ...

SimulationScenario::WetRoad => write!(f, "WET ROAD"),

}

}

}

```
### Passo 3: Implementar Lógica
[src/main.rs](src/main.rs#L49)
```

match scenario {

SimulationScenario::WetRoad => {

scenario_time += DT;

if scenario_time < 2.0 {

throttle = 1.0; // Acelera

} else if scenario_time < 3.0 {

throttle = 0.0;

brake_request = 0.5; // Freio moderado em piso molhado

} else if scenario_time > 8.0 {

brake_request = 0.0;

}

}

// ... outros casos ...

}

```
### Passo 4: Adicionar Tecla de Ativação
[src/main.rs](src/main.rs#L35)
```

KeyCode::Char('5') => {

scenario = SimulationScenario::WetRoad;

scenario_time = 0.0;

simulator.reset();

}

```
### Passo 5: Atualizar Controles (comentário)
[src/main.rs](src/main.rs#L304)
```

output.push_str("■ 1: Manual 2: Emergency ■ 3: HighSpeed 4: Repeated Brake ■\r\n");

output.push_str("■ 5: Wet Road ■ ... ■\r\n");

```
---

## 5. Exemplo: Implementar Função de Análise

### Novo Módulo: metrics.rs
```

// src/metrics.rs

#[derive(Debug, Clone)]

pub struct BrakingMetrics {

pub initial_velocity: f64,

pub final_velocity: f64,

pub stopping_distance: f64,

pub stopping_time: f64,

pub abs_cycles: u32,

pub max_wheel_slip: f64,

pub avg_deceleration: f64,

}

impl BrakingMetrics {

pub fn calculate(

initial_v: f64,

final_v: f64,

distance: f64,

time: f64,

abs_cycles: u32,

max_slip: f64,

) -> Self {

let avg_decel = (initial_v - final_v) / time * 3.6; // m/s²

BrakingMetrics {

initial_velocity: initial_v,

final_velocity: final_v,

stopping_distance: distance,

stopping_time: time,

abs_cycles,

max_wheel_slip: max_slip,

avg_deceleration: avg_decel,

}

}

pub fn print_report(&self) {

println!("■■■

```
| 36 | criterion = { version = "0.5", features = ["html_reports"] } | support logic; impact depends on neighboring blo
| 37 | (blank) | blank separator; no runtime behavior. |
| 38 | [[bench]] | support logic; impact depends on neighboring block and data flow. |
| 39 | name = "solver_bench" | support logic; impact depends on neighboring block and data flow. |
| 40 | harness = false | support logic; impact depends on neighboring block and data flow. |

## File Deep Dive: J1939_CAN_MANUAL.md
Role: project support artifact
Line count: 1565 | Non-empty: 1242 | Symbol count: 111

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Manual Completo: CAN Bus e J1939 | namespace and compile-time linkage |
| 2 | h2 | Guia Definitivo para Veículos Pesados e Maquinário Agrícola | namespace and compile-time linkage |
| 6 | h1 | PARTE 1: CAN BUS — FUNDAMENTOS COMPLETOS | namespace and compile-time linkage |
| 8 | h2 | 1.1 O que é o CAN Bus | namespace and compile-time linkage |
| 12 | h3 | Por que CAN e não Ethernet ou RS-485? | namespace and compile-time linkage |
| 25 | h2 | 1.2 Camada Física (Physical Layer) — ISO 11898-2 | namespace and compile-time linkage |
| 27 | h3 | Diferencial e Topologia | namespace and compile-time linkage |
| 44 | h3 | Níveis de Tensão | namespace and compile-time linkage |
| 60 | h3 | Terminadores | namespace and compile-time linkage |
| 70 | h3 | Velocidades e Comprimentos Máximos | namespace and compile-time linkage |
| 83 | h2 | 1.3 Formato do Frame CAN 2.0 | namespace and compile-time linkage |
| 85 | h3 | Frame de Dados (Data Frame) | namespace and compile-time linkage |
| 114 | h3 | Bits por Frame (CAN 2.0B, 8 bytes dados, sem bit stuffing) | namespace and compile-time linkage |
| 125 | h2 | 1.4 Arbitração — Como Múltiplos Nós Compartilham o Barramento | namespace and compile-time linkage |
| 155 | h2 | 1.5 Tipos de Frames CAN | namespace and compile-time linkage |
| 157 | h3 | Data Frame | namespace and compile-time linkage |
| 160 | h3 | Remote Frame (RTR=1) | namespace and compile-time linkage |
| 163 | h3 | Error Frame | namespace and compile-time linkage |
| 170 | h3 | Overload Frame | namespace and compile-time linkage |
| 175 | h2 | 1.6 Detecção e Confinamento de Erros | namespace and compile-time linkage |
| 177 | h3 | Tipos de Erro Detectados | namespace and compile-time linkage |
| 187 | h3 | Contadores de Erro (ISO 11898-1) | namespace and compile-time linkage |
| 200 | h3 | Estados do Nó (Error Confinement) | namespace and compile-time linkage |
| 220 | h2 | 1.7 Bit Stuffing | namespace and compile-time linkage |
| 233 | h1 | PARTE 2: J1939 — PROTOCOLO COMPLETO | namespace and compile-time linkage |
| 235 | h2 | 2.1 Visão Geral do J1939 | namespace and compile-time linkage |
| 244 | h3 | Stack de Protocolos J1939 | namespace and compile-time linkage |
| 262 | h2 | 2.2 Estrutura do CAN ID de 29 bits no J1939 | namespace and compile-time linkage |
| 271 | h3 | Campos Detalhados | namespace and compile-time linkage |
| 273 | h4 | Priority (Prioridade) — bits 28-26 | namespace and compile-time linkage |
| 286 | h4 | Reserved (R) — bit 25 | namespace and compile-time linkage |
| 289 | h4 | Data Page (DP) — bit 24 | namespace and compile-time linkage |
| 294 | h4 | PF (PDU Format) — bits 23-16 | namespace and compile-time linkage |
| 301 | h4 | PS (PDU Specific) — bits 15-8 | namespace and compile-time linkage |
| 306 | h4 | SA (Source Address) — bits 7-0 | namespace and compile-time linkage |
| 309 | h3 | Cálculo do PGN | namespace and compile-time linkage |
| 319 | h3 | Exemplos Práticos | namespace and compile-time linkage |
| 345 | h2 | 2.3 Endereços de Fonte (Source Addresses) | namespace and compile-time linkage |
| 349 | h3 | Endereços Reservados Importantes | namespace and compile-time linkage |
| 403 | h2 | 2.4 NAME — Identificador Único do Nó (64-bit) | namespace and compile-time linkage |
| 444 | h3 | Exemplos de NAME | namespace and compile-time linkage |
| 459 | h2 | 2.5 Procedimento de Address Claiming (J1939-81) | namespace and compile-time linkage |
| 463 | h3 | Sequência Completa | namespace and compile-time linkage |
| 508 | h3 | Frame de Address Claim | namespace and compile-time linkage |
| 530 | h2 | 2.6 PGN — Parameter Group Number | namespace and compile-time linkage |
| 534 | h3 | Estrutura de um PGN | namespace and compile-time linkage |
| 543 | h3 | Tipos de Transmissão | namespace and compile-time linkage |
| 555 | h2 | 2.7 PGNs Completos — Heavy Machinery | namespace and compile-time linkage |
| 557 | h3 | Grupo Engine (ECM) | namespace and compile-time linkage |
| 559 | h4 | PGN 61444 — EEC1 (Electronic Engine Control 1) | namespace and compile-time linkage |
| 593 | h4 | PGN 61445 — EEC2 (Electronic Engine Control 2) | namespace and compile-time linkage |
| 608 | h4 | PGN 65247 — EEC3 (Electronic Engine Control 3) | namespace and compile-time linkage |
| 620 | h4 | PGN 65262 — ET1 (Engine Temperature 1) | namespace and compile-time linkage |
| 634 | h4 | PGN 65263 — EFL/P1 (Engine Fluid Level/Pressure 1) | namespace and compile-time linkage |
| 649 | h4 | PGN 65270 — IC1 (Inlet/Exhaust Conditions 1) | namespace and compile-time linkage |
| 664 | h4 | PGN 65266 — LFE (Fuel Economy) | namespace and compile-time linkage |
| 677 | h4 | PGN 65253 — HOURS (Engine Hours/Revolutions) | namespace and compile-time linkage |
| 687 | h3 | Grupo Transmission (TCM) | namespace and compile-time linkage |
```

```
| 689 | h4 | PGN 61442 — ETC1 (Electronic Transmission Controller 1) | namespace and compile-time linkage |
| 705 | h4 | PGN 61443 — ETC2 (Electronic Transmission Controller 2) | namespace and compile-time linkage |
| 720 | h3 | Grupo Vehicle (CCVS, VD) | namespace and compile-time linkage |
| 722 | h4 | PGN 65265 — CCVS (Cruise Control/Vehicle Speed) | namespace and compile-time linkage |
| 742 | h4 | PGN 65248 — VD (Vehicle Distance) | namespace and compile-time linkage |
| 752 | h3 | Grupo ISOBUS / Implements | namespace and compile-time linkage |
| 754 | h4 | PGN 65093 — PTO (Power Take-Off Information) | namespace and compile-time linkage |
| 781 | h4 | PGN 65091 — HITCH (Hitch Status, agricultural) | namespace and compile-time linkage |
| 796 | h3 | Grupo Diagnóstico (DM1-DM35) | namespace and compile-time linkage |
| 798 | h4 | PGN 65226 — DM1 (Active Diagnostic Trouble Codes) | namespace and compile-time linkage |
| 823 | h4 | PGN 65227 — DM2 (Previously Active DTCs) | namespace and compile-time linkage |
| 826 | h4 | PGN 65228 — DM3 (Diagnostic Data Clear/Reset) | namespace and compile-time linkage |
| 833 | h4 | PGN 65235 — DM11 (Diagnostic Data Clear) | namespace and compile-time linkage |
| 836 | h4 | PGN 65230 — DM5 (Diagnostic Readiness) | namespace and compile-time linkage |
| 847 | h2 | 2.8 SPN — Suspect Parameter Number | namespace and compile-time linkage |
| 851 | h3 | Estrutura de um SPN | namespace and compile-time linkage |
| 864 | h3 | Cálculo do Valor Físico | namespace and compile-time linkage |
| 887 | h3 | Indicador de Não Disponível | namespace and compile-time linkage |
| 899 | h2 | 2.9 FMI — Failure Mode Identifier | namespace and compile-time linkage |
| 928 | h3 | Exemplos de DTC com FMI | namespace and compile-time linkage |
| 949 | h2 | 2.10 Protocolo de Transporte (J1939-21) | namespace and compile-time linkage |
| 953 | h3 | Protocolo BAM (Broadcast Announce Message) | namespace and compile-time linkage |
| 991 | h3 | Protocolo CMDT (Connection Mode Data Transfer) | namespace and compile-time linkage |
| 1005 | h2 | 2.11 ISOBUS (ISO 11783) — Para Máquinas Agrícolas | namespace and compile-time linkage |
| 1009 | h3 | Working Set Management | namespace and compile-time linkage |
| 1021 | h3 | Task Controller (TC) | namespace and compile-time linkage |
| 1028 | h3 | Virtual Terminal (VT) | namespace and compile-time linkage |
| 1035 | h3 | ECU Address Ranges ISOBUS | namespace and compile-time linkage |
| 1047 | h1 | PARTE 3: DIAGNÓSTICO — COMO FAZER | namespace and compile-time linkage |
| 1049 | h2 | 3.1 Equipamentos Necessários | namespace and compile-time linkage |
| 1051 | h3 | Ferramentas de Hardware | namespace and compile-time linkage |
| 1061 | h3 | Software | namespace and compile-time linkage |
| 1071 | h2 | 3.2 Verificação da Camada Física | namespace and compile-time linkage |
| 1073 | h3 | Passo 1: Verificar Resistência do Barramento | namespace and compile-time linkage |
| 1086 | h3 | Passo 2: Verificar Tensões em Operação | namespace and compile-time linkage |
| 1101 | h3 | Passo 3: Verificar com Osciloscópio | namespace and compile-time linkage |
| 1129 | h2 | 3.3 Diagnóstico de DTCs | namespace and compile-time linkage |
| 1131 | h3 | Leitura de DTCs via J1939 | namespace and compile-time linkage |
| 1157 | h3 | Interpretação dos DTCs | namespace and compile-time linkage |
| 1199 | h3 | Processo de Diagnóstico | namespace and compile-time linkage |
| 1230 | h2 | 3.4 Programação e Reprogramação de ECUs | namespace and compile-time linkage |
| 1232 | h3 | Flashing via CAN (J1939-21) | namespace and compile-time linkage |
| 1253 | h3 | Calibração de Parâmetros (EEPROM Write) | namespace and compile-time linkage |
| 1270 | h1 | PARTE 4: IMPLEMENTAÇÃO PRÁTICA | namespace and compile-time linkage |
| 1272 | h2 | 4.1 Decodificação de Frame J1939 em Rust | namespace and compile-time linkage |
| 1309 | h2 | 4.2 Extraindo Bits de um SPN | namespace and compile-time linkage |
| 1339 | h2 | 4.3 Transmitindo Frame J1939 | namespace and compile-time linkage |
| 1367 | h2 | 4.4 Implementando Address Claiming | namespace and compile-time linkage |
| 1426 | h1 | PARTE 5: REFERÊNCIA RÁPIDA | namespace and compile-time linkage |
| 1428 | h2 | 5.1 PGNs Mais Usados em Heavy Machinery | namespace and compile-time linkage |
| 1460 | h2 | 5.2 SPNs Mais Importantes | namespace and compile-time linkage |
| 1493 | h2 | 5.3 Tabela de Troubleshooting Rápido | namespace and compile-time linkage |
| 1509 | h1 | PARTE 6: GLOSSÁRIO | namespace and compile-time linkage |

### Content Snapshot
```

# Manual Completo: CAN Bus e J1939

## Guia Definitivo para Veículos Pesados e Maquinário Agrícola

---

# PARTE 1: CAN BUS — FUNDAMENTOS COMPLETOS

# 1.1 O que é o CAN Bus

O **Controller Area Network (CAN)** é um protocolo de comunicação serial desenvolvido pela Bosch em 1986, projetado especificamente para ambientes automotivos e industriais ruidosos. Tornou-se o padrão dominante para comunicação entre ECUs (Unidades de Controle Eletrônico) em veículos.

## Por que CAN e não Ethernet ou RS-485?

```
| Característica | CAN | RS-485 | Ethernet |
|---------------|-----|--------|---------|
| Topologia | Multi-drop bus | Multi-drop | Estrela/Switch |
| Arbitração | Sem destruição de dados | Colisão = dados perdidos | CSMA/CD |
| Tempo real | Determinístico | Semi | Não determinístico |
| Tolerância a falha | Alta (bus-off, error frames) | Baixa | Baixa |
| Taxa típica | 125k–1M bps | 100k–10M bps | 10M–10G bps |
| Custo (cabling) | Muito baixo (2 fios) | Baixo | Alto |
```

---

# 1.2 Camada Física (Physical Layer) — ISO 11898-2

## Diferencial e Topologia

O CAN usa sinalização diferencial em dois fios:

- **CAN_H** (CAN High)
- **CAN_L** (CAN Low)

```
          Terminador         Terminador
          120 Ω              120 Ω
    ■■■■■■■■■■■■■                ■■■■■■■■■■■■■
    ■          ■                ■          ■
ECU1■CAN_H ■■■■■■■■■■■■■
```

### Velocidades e Comprimentos Máximos

```
| Velocidade | Comp. Máximo | Aplicação |
|-----------|-------------|-----------|
| 1 Mbit/s  | 40 m        | ECU interno (J1939 não usa) |
| 500 kbit/s| 100 m       | HS-CAN automotivo (J1939 padrão) |
| 250 kbit/s| 250 m       | J1939 agrícola/off-highway |
| 125 kbit/s| 500 m       | LowSpeed CAN, carroçaria |
| 50 kbit/s | 1000 m      | Indústria, longos cabos |
| 25 kbit/s | 2000 m      | Aplicações especiais |
```

---

## 1.3 Formato do Frame CAN 2.0

### Frame de Dados (Data Frame)

```
┌──────────────────────────────────────────────────────────────────────────────┐
│ SOF │ ID │ RTR│ Control (DLC+IDE) │DATA │CRC │ ACK │ EOF │IFS │
└──────────────────────────────────────────────────────────────────────────────┘
  1b   11b  1b        6b            0-64b  16b   2b    7b   3b
```

Para CAN 2.0B (29-bit Extended ID, usado pelo J1939):

```
┌──────────────────────────────────────────────────────────────────────────────────┐
│ SOF │ ID[28-18]│SRR│IDE│ID[17-0]│RTR│r1 │DLC │DATA│CRC │ACK │EOF│
└──────────────────────────────────────────────────────────────────────────────────────┘
  1b    11b     1b   1b   18b     1b   1b   4b   0-64b 15b  2b  7b
```

**Campos:**

- **SOF** (Start Of Frame): Bit dominante — indica início do frame
- **ID**: Identificador (11-bit padrão ou 29-bit estendido)
- **RTR** (Remote Transmission Request): 0=dados, 1=request remoto
- **IDE** (Identifier Extension): 0=11-bit, 1=29-bit
- **DLC** (Data Length Code): 0-8 bytes de dados
- **DATA**: Payload de 0 a 8 bytes
- **CRC**: CRC-15 para detecção de erros
- **ACK**: Todos nós receptores puxam para dominante se receberam OK
- **EOF**: 7 bits recessivos — fim do frame
- **IFS** (Inter-Frame Space): 3 bits mínimo entre frames

### Bits por Frame (CAN 2.0B, 8 bytes dados, sem bit stuffing)

```
SOF:1 + ID29:29 + outros headers:6 + DLC:4 + DATA:64 + CRC:16 + ACK:2 + EOF:7 + IFS:3 = 132 bits
```

Com bit stuffing médio: ≈ 148 bits por frame.

**A 500 kbit/s**: ~500000/148 ≈ **3378 frames/segundo máximo**.

---

## 1.4 Arbitração — Como Múltiplos Nós Compartilham o Barramento

O CAN usa **CSMA/CD com arbitração não-destrutiva baseada em prioridade**.

```
Processo de transmissão:
1. Nó aguarda barramento recessivo (IFS + Bus Idle)
2. Nó começa a transmitir bit a bit
3. Enquanto transmite, MONITORA o barramento
4. Se o bit enviado = bit lido → continua
5. Se enviou RECESSIVO mas leu DOMINANTE → PERDEU arbitração
6. Nó perdedor para transmissão imediatamente (sem dano ao frame)
7. Nó vencedor continua até o fim do frame
```

**Exemplo de arbitração:**

```
Tempo →     b1  b2  b3  b4  b5  b6  b7  ...
Nó A (ID=0x0CF004):  0   0   1   1   0   0   0  ← Transmite
Nó B (ID=0x18FEF0):  0   1   ...                 ← Transmite b2=1, barramento=0 → perde!
Barramento:          0   0   ...                 ← Bit 2 dominante (Nó A vence)
```

**Regra**: ID **menor** = **maior prioridade**.

- 0x0000000 = máxima prioridade
- 0x1FFFFFFF = mínima prioridade

---

## 1.5 Tipos de Frames CAN

### Data Frame

Frame normal com dados.

### Remote Frame (RTR=1)

Solicita que outro nó transmita dados. Não contém payload.

### Error Frame

Gerado automaticamente pelo hardware quando detecta erro:

```
■■■ Error Flag: 6 bits dominantes (ativo) ou recessivos (passivo)
■■■ Error Delimiter: 8 bits recessivos
```

### Overload Frame

Indica que o receptor precisa de mais tempo antes do próximo frame.

---

## 1.6 Detecção e Confinamento de Erros

### Tipos de Erro Detectados

```
| Tipo | Descrição | Detectado por |
|------|-----------|--------------|
| **Bit Error** | Nó monitora e vê bit diferente do enviado | Transmissor |
| **Stuff Error** | 6+ bits consecutivos do mesmo nível | Todos |
| **CRC Error** | CRC calculado ≠ CRC recebido | Receptor |
| **Form Error** | Campo EOF/Ack com bit dominante | Todos |
| **ACK Error** | Nenhum nó reconheceu o frame | Transmissor |
```

### Contadores de Erro (ISO 11898-1)

Cada nó mantém dois contadores:

- **TEC** (Transmit Error Counter): Incrementa em erros TX
- **REC** (Receive Error Counter): Incrementa em erros RX

```
Regras de incremento/decremento:
TX Error:  TEC += 8  (por cada erro de transmissão)
RX Error:  REC += 1  (por cada erro de recepção)
Sucesso:   TEC -= 1, REC -= 1  (por frame sem erro)
```

### Estados do Nó (Error Confinement)

```
                TEC < 128 AND REC < 128
                        ■
                    ERROR ACTIVE  ■■■■■■■■■■■■■■■■■■■■■■■■
            (envia Active Error Flag)                 ■
                        ■ TEC ≥ 128 OR REC ≥ 128       ■ TEC diminui
                        ▼                              ■ < 128
                    ERROR PASSIVE ■■■■■■■■■■■■■■■■■■■■■■■■■■■
            (envia Passive Error Flag)                 ■■
                        ■ TEC ≥ 256                   ■■
                        ▼                             ■■
                       BUS-OFF  ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■
            (nó desconecta do barramento)
            Recovery: 128 × 11 bits recessivos
```

---

## 1.7 Bit Stuffing

... truncated 1345 lines for this snapshot in markdown volume ...

```
## File Deep Dive: PROJECT_SUMMARY.md
Role: project support artifact
Line count: 381 | Non-empty: 299 | Symbol count: 53

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | ■ Sumário Completo do Projeto - Sistema de Freio Autônomo com ABS | namespace and compile-time linkage |
| 3 | h2 | ■ O que você tem agora | namespace and compile-time linkage |
| 9 | h2 | ■ Estrutura de Arquivos | namespace and compile-time linkage |
| 31 | h2 | ■ Documentação Incluída | namespace and compile-time linkage |
| 33 | h3 | 1. **README.md** (este arquivo) | namespace and compile-time linkage |
| 41 | h3 | 2. **TESTING.md** | namespace and compile-time linkage |
| 48 | h3 | 3. **ANALYSIS.md** | namespace and compile-time linkage |
| 57 | h3 | 4. **TUTORIALS.md** | namespace and compile-time linkage |
| 64 | h3 | 5. **ARCHITECTURE.md** | namespace and compile-time linkage |
| 75 | h2 | ■ Recursos Principais | namespace and compile-time linkage |
| 77 | h3 | Sistema de Simulação (lib.rs) | namespace and compile-time linkage |
| 93 | h3 | Interface Visual (main.rs) | namespace and compile-time linkage |
| 107 | h2 | ■ Como Começar | namespace and compile-time linkage |
| 109 | h3 | 1. Compilação Inicial | namespace and compile-time linkage |
| 116 | h3 | 2. Execução | namespace and compile-time linkage |
| 122 | h3 | 3. Testar Cenários | namespace and compile-time linkage |
| 131 | h2 | ■ Características Técnicas | namespace and compile-time linkage |
| 133 | h3 | Performance | namespace and compile-time linkage |
| 139 | h3 | Física | namespace and compile-time linkage |
| 145 | h3 | Fidelidade | namespace and compile-time linkage |
| 153 | h2 | ■ Dados Exportáveis | namespace and compile-time linkage |
| 169 | h2 | ■ Casos de Uso | namespace and compile-time linkage |
| 171 | h3 | Educacional | namespace and compile-time linkage |
| 177 | h3 | Pesquisa | namespace and compile-time linkage |
| 183 | h3 | Protótipo | namespace and compile-time linkage |
| 191 | h2 | ■ Tecnologias Utilizadas | namespace and compile-time linkage |
```

```
| 193 | h3 | Linguagem | namespace and compile-time linkage |
| 196 | h3 | Dependências | namespace and compile-time linkage |
| 202 | h3 | Padrões de Design | namespace and compile-time linkage |
| 210 | h2 | ■ Validação e Testes | namespace and compile-time linkage |
| 212 | h3 | Testes Inclusos | namespace and compile-time linkage |
| 221 | h3 | Validação Manual | namespace and compile-time linkage |
| 230 | h2 | ■ Roadmap de Aprendizado | namespace and compile-time linkage |
| 232 | h3 | Iniciante | namespace and compile-time linkage |
| 238 | h3 | Intermediário | namespace and compile-time linkage |
| 244 | h3 | Avançado | namespace and compile-time linkage |
| 250 | h3 | Profissional | namespace and compile-time linkage |
| 258 | h2 | ■ Extensões Planejadas | namespace and compile-time linkage |
| 260 | h3 | Curto Prazo (v0.2) | namespace and compile-time linkage |
| 266 | h3 | Médio Prazo (v0.3) | namespace and compile-time linkage |
| 272 | h3 | Longo Prazo (v1.0) | namespace and compile-time linkage |
| 280 | h2 | ■ Suporte Técnico | namespace and compile-time linkage |
| 282 | h3 | Problemas Comuns | namespace and compile-time linkage |
| 286 | h1 | Solução: | namespace and compile-time linkage |
| 293 | h1 | Verificar: | namespace and compile-time linkage |
| 307 | h2 | ■ Referências Externas | namespace and compile-time linkage |
| 309 | h3 | Padrões de Projeto | namespace and compile-time linkage |
| 313 | h3 | Engenharia Automotiva | namespace and compile-time linkage |
| 318 | h3 | Rust | namespace and compile-time linkage |
| 325 | h2 | ■ Licença | namespace and compile-time linkage |
| 338 | h2 | ■■■ Informações do Projeto | namespace and compile-time linkage |
| 355 | h2 | ■ Checklist Final | namespace and compile-time linkage |
| 369 | h2 | ■ Próximos Passos | namespace and compile-time linkage |

### Content Snapshot
```

# ■ Sumário Completo do Projeto - Sistema de Freio Autônomo com ABS

## ■ O que você tem agora

Um **simulador funcional e interativo** de sistema autônomo de freio com ABS desenvolvido em Rust, com painel visual em tempo real e suporte a múltiplos cenários de teste.

---

## ■ Estrutura de Arquivos

```
AutoBreaking/
■■■ Cargo.toml                # Configuração do projeto e dependências
■■■ Cargo.lock                # Lock de versões (gerado)
■■■ README.md                 # ■ Documentação principal
■■■ TESTING.md                # ■ Guia de testes rápidos
■■■ ANALYSIS.md               # ■ Análise técnica aprofundada
■■■ TUTORIALS.md              # ■ Tutoriais para iniciante→profissional
■■■ ARCHITECTURE.md           # ■■ Arquitetura e como estender
■■■ src/
■   ■■■ lib.rs                # Biblioteca: Simulação physics + ABS
■   ■■■ main.rs               # Interface: Terminal UI interativa
■   ■■■ (gerados: build artefatos)
■■■ target/                   # Compilados (não commitar)
    ■■■ debug/                # Builds debug
    ■■■ release/              # Builds otimizados
```

---

## ■ Documentação Incluída

### 1. **README.md** (este arquivo)

- ■ Características principais
- ■ Controles da simulação
- ■ Arquitetura técnica
- ■ Casos de uso educacionais
- ■ Requisitos e dependências
- ■ Futuras melhorias

### 2. **TESTING.md**

- ■ Verificação de compilação
- ■ 4 cenas de teste recomendadas
- ■ Métricas para análise
- ■ Observações de engenharia
- ■ Checklist de validação

### 3. **ANALYSIS.md**

- ■ Equações diferenciais (7 seções)
- ■ Algoritmo de controle ABS
- ■ Modelos de sensor
- ■ Análise de performance
- ■ Comparação com sistemas reais
- ■ Validação de comportamento
- ■ Fórmulas auxiliares

### 4. **TUTORIALS.md**

- ■ Iniciante: Primeiro Contato (5 etapas)
- ■ Intermediário: Análise de Cenários (3 testes)
- ■ Avançado: Interpretação de Dados (4 análises)
- ■ Profissional: Estudo de Caso (4 casos)
- ■ Referências de estudo

### 5. **ARCHITECTURE.md**

- ■ Diagrama de arquitetura
- ■ Estrutura de módulos
- ■ Fluxo de dados
- ■ Detalhes de implementação
- ■ Interface terminal

- ■ 5 formas de estender o sistema
- ■ Roadmap de desenvolvimento

---

## ■ Recursos Principais

### Sistema de Simulação (lib.rs)

```
- SpeedSensor         Sensor com ruído realístico
- Wheel               Dinâmica individual de roda
- BrakeSystem         4 rodas + pressões
- ABSController       Algoritmo de controle ABS
- VehicleSimulator    Integração completa
```

**Componentes técnicos**:

- ■ Detecção de travamento
- ■ Modulação ABS (8 Hz)
- ■ Dinâmica não-linear
- ■ Sensor com ruído gaussiano
- ■ Integração numérica Euler

### Interface Visual (main.rs)

```
- Painel em tempo real (60 FPS)
- Status do veículo (velocidade, aceleração, tempo)
- Controle de entradas (throttle, brake)
- Pressão individual por roda
- Estado de cada roda (■ ■ ■)
- Histórico de velocidade (gráfico)
- 4 cenários automáticos
- Controles interativos (teclado)
```

---

## ■ Como Começar

### 1. Compilação Inicial

```
cd AutoBreaking
cargo build
```

**Tempo**: ~30-60 segundos (primeira vez)

### 2. Execução

```
cargo run
```

**Resultado**: Interface visual preenche a tela

### 3. Testar Cenários

- Pressione **1-4** para diferentes modos
- Use **↑↓←→** para controle manual

- Pressione **SPACE** para pausar
- Pressione **R** para resetar
- Pressione **Q** para sair

---

## ■ Características Técnicas

### Performance

- **Taxa de atualização**: 60 FPS (16ms por frame)
- **Latência de resposta**: 0-16ms
- **Overhead computacional**: Mínimo (<1ms/ciclo)
- **Uso de memória**: ~2 KB

### Física

- **Integração numérica**: Euler de 1ª ordem
- **Erro acumulado**: ±5% por segundo
- **Máxima velocidade**: 200 km/h
- **Desaceleração máxima**: 10 m/s²

### Fidelidade

- **Frequência ABS**: 8 Hz (realista)
- **Ruído do sensor**: ±0.5 km/h σ (realístico)
- **Limiar de travamento**: 5 km/h (configurável)
- **Modulação ABS**: 30-90% (típico)

---

## ■ Dados Exportáveis

Cada simulação fornece:

```
- Velocidade instantânea (km/h)
- Leitura do sensor (com ruído)
- Aceleração (m/s²)
- Pressão de freio (4 rodas)
- Estado das rodas (4 estados)
- Ciclos ABS (contador)
- Tempo decorrido
- Histórico de velocidade (100 amostras)
```

---

## ■ Casos de Uso

### Educacional

- ■ Entender dinâmica de freio
- ■ Aprender sobre controle em tempo real
- ■ Estudar sistemas automotivos
- ■ Analisar algoritmos de controle

### Pesquisa

- ■ Validar modelos matemáticos
- ■ Testar algoritmos ABS novos
- ■ Comparar estratégias de controle
- ■ Benchmarking de performance

### Protótipo

- ■ Desenvolvimento de software automotivo
- ■ Validação de lógica ECU
- ■ Integração com hardware simulado
- ■ Testes de segurança funcional

---

## ■ Tecnologias Utilizadas

### Linguagem

- **Rust 1.70+** - Segurança de memória sem GC

### Dependências

- **crossterm 0.27** - UI em terminal com suporte ANSI
- **chrono 0.4** - Manipulação de datas/horas
- **serde 1.0** - Serialização de dados
- **serde_json 1.0** - Formato JSON

### Padrões de Design

- **State Pattern** - Estados das rodas
- **Strategy Pattern** - Cenários de simulação
- **Observer Pattern** - Atualização de UI
- **Factory Pattern** - Criação de simulador

---

## ■ Validação e Testes

### Testes Inclusos

```
cargo test
```

Verifica:

- ■ Criação de simulador
- ■ Aceleração
- ■ Cálculos de dinâmica

... truncated 161 lines for this snapshot in markdown volume ...

```
## File Deep Dive: README.md
Role: project support artifact
Line count: 290 | Non-empty: 221 | Symbol count: 28

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | SimuBench | namespace and compile-time linkage |
| 12 | h2 | Visao Rapida | namespace and compile-time linkage |
| 22 | h2 | Comece Aqui | namespace and compile-time linkage |
| 24 | h3 | 1. Rodar em simulacao | namespace and compile-time linkage |
| 30 | h3 | 2. Rodar o binario principal explicitamente | namespace and compile-time linkage |
| 36 | h3 | 3. Rodar o runner headless | namespace and compile-time linkage |
| 42 | h3 | 4. Validar qualidade | namespace and compile-time linkage |
| 50 | h2 | O Que Voce Vai Ver Na UI | namespace and compile-time linkage |
| 54 | h3 | Operacao | namespace and compile-time linkage |
| 65 | h3 | Diagnostico | namespace and compile-time linkage |
| 76 | h2 | Fluxo Recomendado | namespace and compile-time linkage |
| 88 | h2 | Principais Capacidades | namespace and compile-time linkage |
| 90 | h3 | Simulacao multi-ECU | namespace and compile-time linkage |
| 98 | h3 | CAN, J1939 e UDS | namespace and compile-time linkage |
| 113 | h3 | Sensores e autonomia | namespace and compile-time linkage |
| 132 | h3 | Leak Physics Lab | namespace and compile-time linkage |
| 145 | h2 | Leak Lab Em 30 Segundos | namespace and compile-time linkage |
| 147 | h3 | O que ele faz | namespace and compile-time linkage |
| 155 | h3 | Fluxo rapido | namespace and compile-time linkage |
| 163 | h3 | CSV esperado para calibracao | namespace and compile-time linkage |
| 181 | h2 | Modo Live e Seguranca | namespace and compile-time linkage |
| 185 | h3 | Flags importantes | namespace and compile-time linkage |
| 202 | h3 | Gates de escrita fisica | namespace and compile-time linkage |
| 214 | h2 | Binarios | namespace and compile-time linkage |
| 224 | h2 | Estrutura Rapida do Repositorio | namespace and compile-time linkage |
| 248 | h2 | Qualidade e Testes | namespace and compile-time linkage |
| 268 | h2 | Documentacao Auxiliar | namespace and compile-time linkage |
| 276 | h2 | Estado Atual | namespace and compile-time linkage |

### Content Snapshot
```

# SimuBench

SimuBench e uma bancada visual de simulacao, diagnostico e engenharia para ECUs de maquinas pesadas, escrita em Rust.

Em um unico aplicativo, o projeto junta:

- simulacao multi-ECU em tempo real;
- painel desktop com tabs operacionais e de diagnostico;
- CAN/J1939, UDS e captura ECM Live;
- Leak Physics Lab com predicao, Monte Carlo e calibracao via CSV de bancada;
- camada de I/O com politicas de seguranca para uso em hardware real.

## Visao Rapida

```
| Area | O que entrega |
|---|---|
| Simulacao | ECM, TCM, ABS/ESP/TCS, BCM, HCM, VCM, sensores e telematica rodando juntos |
| Diagnostico | Trace CAN, sinais J1939, faults, boot, UDS e rede ECU |
| Dados live | Detect, connect, retrieve e export de dados ECM |
| Engenharia | Leak Lab para risco, ruptura, tuning e calibracao automatica |
| Seguranca | Allowlist, rate limit, dry-run e gates explicitos para escrita |
```

## Comece Aqui

### 1. Rodar em simulacao

```
cargo run -- --hw-mode=sim
```

### 2. Rodar o binario principal explicitamente

```
cargo run --bin auto_breaking -- --hw-mode=sim
```

### 3. Rodar o runner headless

```
cargo run --bin simulator_cli -- --seed 7 --steps 20000 --model reduced
```

### 4. Validar qualidade

```
cargo check
cargo clippy --workspace --all-targets
cargo test --workspace
```

## O Que Voce Vai Ver Na UI

O app foi dividido em duas frentes:

### Operacao

- `Cluster`: velocidade, RPM, marcha, lamps e sinais principais
- `Engine`: torque, carga, temperaturas, aftertreatment e DTCs ativos
- `Implements`: PTO, hitch, loader, auxiliares e hidraulica
- `Sensors`: GPS, IMU, radar, lidar, camera e coerencia de percepcao
- `Autonomous`: ACC, AEB, LKA, TJA e saidas de comando
- `V2X`: V2V, SPaT, work zones, telematica e OTA
- `Leak Lab`: laboratorio visual para leak/risco/ruptura
- `Plots`: comparacao temporal rapida dos principais canais

### Diagnostico

- `CAN Bus`: sinais, trace e estado dos barramentos
- `Events`: historico de eventos e correlacao temporal
- `ECU Net`: rede de ECUs, presenca e comportamento de boot
- `Faults`: injecao de falhas e observacao de DTCs/DM1
- `Boot`: sequencia de ignicao, interlocks e readiness

- `UDS`: console de servicos ISO 14229
- `ECM Live`: fluxo real/simulado de captura de parametros
- `Help`: leitura guiada do produto e da operacao

## Fluxo Recomendado

Se voce esta chegando agora, use esta ordem:

1. `Help` para entender a navegacao.
2. `Cluster` e `Engine` para validar se a maquina liga, troca marcha e se move.
3. `CAN Bus` e `Events` para ver rede e correlacao temporal.
4. `Faults` para injetar falha controlada e acompanhar DTC/DM1.
5. `UDS` para exercitar sessao, seguranca, leitura e limpeza.
6. `ECM Live` para capturar/exportar sinais.
7. `Leak Lab` para engenharia, risco e calibracao.

## Principais Capacidades

### Simulacao multi-ECU

O orchestrator central integra os modulos de powertrain, chassis, hidraulica, sensores, telematica e diagnostico.

Arquivos principais:

- [src/lib.rs](src/lib.rs)
- [src/main.rs](src/main.rs)

### CAN, J1939 e UDS

Cobertura da pilha de rede:

- gateway e multi-bus CAN;
- estados de saude e injecao de erro;
- decode J1939;
- servidor UDS para ECM e TCM;
- log visual no desktop.

Arquivos principais:

- [src/can_gateway.rs](src/can_gateway.rs)
- [src/can_network.rs](src/can_network.rs)
- [src/j1939.rs](src/j1939.rs)
- [src/uds.rs](src/uds.rs)

### Sensores e autonomia

O projeto inclui sensores sinteticos e fusao para cenarios AD:

- GPS
- IMU
- radar
- lidar
- camera
- fusao de sensores
- controlador autonomo

Arquivos principais:

- [src/gps.rs](src/gps.rs)
- [src/imu.rs](src/imu.rs)
- [src/radar.rs](src/radar.rs)
- [src/lidar.rs](src/lidar.rs)
- [src/camera.rs](src/camera.rs)
- [src/autonomous.rs](src/autonomous.rs)

### Leak Physics Lab

Ambiente de engenharia para circuitos hidraulicos e vedacao:

- runtime por circuito;
- predicao por horizonte e `dt`;
- Monte Carlo;
- export runtime/prediction/catalogos;
- calibracao automatica com CSV de bancada;
- ASCII CAD e visualizacoes de apoio.

Arquivo principal:

- [src/leak_physics.rs](src/leak_physics.rs)

## Leak Lab Em 30 Segundos

### O que ele faz

- mostra leak atual por circuito;
- estima risco de ruptura;
- projeta cenarios;
- roda Monte Carlo;
- calibra coeficientes com dados reais.

### Fluxo rapido

1. Selecione um circuito.
2. Ajuste parametros manuais, se necessario.
3. Clique em `Aplicar + Rodar Cenario` ou `Aplicar + Rodar Monte Carlo`.
4. Leia o `Scenario Ranking`.
5. Use export CSV/JSON para rastreabilidade.

### CSV esperado para calibracao

```
timestamp_s,circuit_name,pressure_bar,delta_p_bar,temp_c,cycles_per_s,duty_01,fluid_density_kg_m3,measured_leak_lpm,ob
```

Campos:

- `timestamp_s`: tempo da amostra
- `circuit_name`: nome do circuito
- `pressure_bar`: pressao instantanea
- `delta_p_bar`: diferencial de pressao
- `temp_c`: temperatura do fluido
- `cycles_per_s`: frequencia de ciclagem
- `duty_01`: duty entre `0` e `1`
- `fluid_density_kg_m3`: densidade do fluido
- `measured_leak_lpm`: vazao medida
- `observed_rupture`: ruptura observada (`true/false`)

## Modo Live e Seguranca

O projeto foi desenhado para ser seguro por padrao quando ha I/O real.

### Flags importantes

```
--hw-mode=sim|live
--vendor-name=cat_comm
--serial-port
--serial-baud
--can-if
--enable-write
--allowlist
--noninteractive-approved
--dry-run / --dry-run=false
--rate-limit-global
--rate-limit-per-id
--log-dir
```

### Gates de escrita fisica

- `enable-write`
- allowlist valida
- `noninteractive-approved`
- `dry-run` explicitamente desabilitado

Arquivos principais:

- [src/io/hw.rs](src/io/hw.rs)
- [src/io/live_runner.rs](src/io/live_runner.rs)
- [docs/ECM-Data.md](docs/ECM-Data.md)

## Binarios

O repositorio possui dois executaveis:

- `auto_breaking`: aplicacao desktop principal
- `simulator_cli`: runner headless para reproducao e carga

... truncated 70 lines for this snapshot in markdown volume ...

```
## File Deep Dive: TESTING.md
Role: project support artifact
Line count: 254 | Non-empty: 200 | Symbol count: 32

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | ■ Guia de Teste Rápido - Sistema Autônomo de Freio com ABS | namespace and compile-time linkage |
| 3 | h2 | Verificação Inicial de Compilação | namespace and compile-time linkage |
| 13 | h2 | ■ Cenas de Teste Recomendadas | namespace and compile-time linkage |
| 15 | h3 | Teste 1: Freio Manual Progressivo (3-5 minutos) | namespace and compile-time linkage |
| 34 | h3 | Teste 2: Freio de Emergência (2-3 minutos) | namespace and compile-time linkage |
| 51 | h3 | Teste 3: Alta Velocidade (3-4 minutos) | namespace and compile-time linkage |
| 66 | h3 | Teste 4: Frenagens Repetidas (4-5 minutos) | namespace and compile-time linkage |
| 83 | h2 | ■ Métricas para Análise | namespace and compile-time linkage |
| 85 | h3 | Velocidade | namespace and compile-time linkage |
| 90 | h3 | Estado das Rodas | namespace and compile-time linkage |
| 97 | h3 | Pressão de Freio | namespace and compile-time linkage |
| 102 | h3 | Ciclos ABS | namespace and compile-time linkage |
| 109 | h2 | ■ Observações de Engenharia | namespace and compile-time linkage |
| 111 | h3 | Comportamento Normal ■ | namespace and compile-time linkage |
| 120 | h3 | Comportamento Anômalo ■■ | namespace and compile-time linkage |
| 130 | h2 | ■ Dados Técnicos para Análise | namespace and compile-time linkage |
| 132 | h3 | Parâmetros de Simulação | namespace and compile-time linkage |
| 142 | h3 | Limites de Detecção | namespace and compile-time linkage |
| 149 | h3 | Dinâmica de Roda | namespace and compile-time linkage |
| 158 | h2 | ■ Gráfico de Teste: Velocidade vs Tempo | namespace and compile-time linkage |
| 160 | h3 | Teste de Freio Normal (sem ABS) | namespace and compile-time linkage |
| 176 | h3 | Teste com ABS Ativo | namespace and compile-time linkage |
| 194 | h2 | ■ Checklist de Validação | namespace and compile-time linkage |
| 214 | h2 | ■ Troubleshooting | namespace and compile-time linkage |
| 216 | h3 | "Cannot find module" ao compilar | namespace and compile-time linkage |
| 218 | h1 | Solução: | namespace and compile-time linkage |
| 223 | h3 | Interface não renderiza | namespace and compile-time linkage |
| 225 | h1 | Verificar terminal: | namespace and compile-time linkage |
| 230 | h1 | Solução: | namespace and compile-time linkage |
| 234 | h3 | Comportamento imprevisível | namespace and compile-time linkage |
| 236 | h1 | Reset do estado: | namespace and compile-time linkage |
| 244 | h2 | ■ Recursos Adicionais | namespace and compile-time linkage |

### Content Snapshot
```

# ■ Guia de Teste Rápido - Sistema Autônomo de Freio com ABS

## Verificação Inicial de Compilação

■ **Status**: Sistema compilado e executado com sucesso!

```
cargo build   # Compilação debug
cargo run     # Execução
cargo test    # Testes unitários
```

## ■ Cenas de Teste Recomendadas

### Teste 1: Freio Manual Progressivo (3-5 minutos)

**Objetivo**: Entender como o ABS reage a diferentes níveis de frenagem

1. Pressione `1` para **Modo Manual**

2. Pressione **↑** 5 vezes para acelerar até ~50 km/h

3. Pressione **→** lentamente aumentando freio de 10% em 10%

4. Observe:

- ✓ Quando velocidade das rodas começa a cair (Skidding detectado)

- ✓ Quando ABS é ativado (■ ABS nos status das rodas)

- ✓ Ciclos ABS aumentando

- ✓ Pressão pulsando entre 30-90%

**Resultado Esperado**:

- Em ~30-40% de freio, ABS é ativado
- Rodas mantêm rotação enquanto desacelerando
- Distância de parada é otimizada

---

### Teste 2: Freio de Emergência (2-3 minutos)

**Objetivo**: Validar sistema em cenário crítico

1. Pressione `2` para **Emergency Brake**

2. Observe automaticamente:

- Aceleração de 0 a ~100 km/h (3 segundos)

- Aplicação de freio máximo (100%)

- ABS ativado imediatamente

- Múltiplos ciclos ABS

**Resultado Esperado**:

- Parada completa em ~5-6 segundos
- 8-12 ciclos de ABS
- Nenhuma roda deve travar (todas em ■ ou ■)

---

### Teste 3: Alta Velocidade (3-4 minutos)

**Objetivo**: Testar limites do sistema em alta velocidade

1. Pressione `3` para **High Speed**
2. Sistema acelera até velocidade máxima (~160-180 km/h)
3. Aplica frenagem moderada
4. Observe sensibilidade do ABS

**Resultado Esperado**:

- ABS ativo por mais tempo (maior diferença de velocidade)
- Mais ciclos ABS necessários
- Comportamento estável em alta velocidade

---

### Teste 4: Frenagens Repetidas (4-5 minutos)

**Objetivo**: Validar robustez do sistema

1. Pressione `4` para **Repeated Braking**
2. Sistema realiza 5 ciclos de:

- Aceleração moderada
- Frenagem
- Recuperação

3. Analise consistência

**Resultado Esperado**:

- Ciclos ABS consistentes entre frenagens
- Sem travamento entre ciclos
- Distância de parada previsível

---

## ■ Métricas para Análise

### Velocidade

- **Real**: Velocidade atual do veículo
- **Sensor**: Leitura com ruído (±0.5 km/h)
- **Diferença**: Desvio padrão do sensor

### Estado das Rodas

```
■ ROLLING    = Rodando normalmente (sem freio)
```

```
■ ABS        = Sistema ABS atuando (modulando pressão)
■ SKIDDING   = Roda travada (perigo!)
```

### Pressão de Freio

- **Escala**: 0% (sem freio) a 100% (freio máximo)
- **Com ABS**: Pulsa entre 30-90%
- **Sem ABS**: Constante

### Ciclos ABS

- **Incrementa**: A cada pulsação (8 Hz)
- **Indicador**: Efetividade do sistema
- **Esperado**: 1-2 ciclos por 0.1 segundo

---

## ■ Observações de Engenharia

### Comportamento Normal ■

```
- Aceleração linear
- Freio sem travamento com ABS
- Rodas dianteiras mais sensíveis
- Pulsação regular a 8 Hz
- Velocidade sensor ~ velocidade real
```

### Comportamento Anômalo ■■

```
- Roda vermelha (■ SKIDDING) por >1 segundo
- Pressão não pulsando corretamente
- Diferença velocidade roda/veículo >15 km/h
- Aceleração negativa sem freio
```

---

## ■ Dados Técnicos para Análise

### Parâmetros de Simulação

```
Frequência ABS:         8 Hz (período: 0.125s)
Taxa de atualização:    60 FPS (dt: 0.016s)
Velocidade máxima:      200 km/h
Aceleração máxima:      5 m/s²
Desaceleração máxima:   10 m/s²
Ruído do sensor:        0.5 km/h σ
```

### Limites de Detecção

```
Travamento ativado:     Δv > 5 km/h E pressão > 0.3
Freio considerado:      Requisição > 0.1
Rodas afetadas:         Todas as 4 igualmente
```

### Dinâmica de Roda

```
Desaceleração: brake_force * 10.0 m/s²
Velocidade mínima: 0 km/h (sempre ≥ 0)
Atualização: Euler de 1ª ordem
```

---

## ■ Gráfico de Teste: Velocidade vs Tempo

### Teste de Freio Normal (sem ABS)

```
200 km/h ■
         ■     ■■
150 km/h ■    ■  ■
         ■   ■    ■
100 km/h ■  ■      ■
         ■ ■        ■
 50 km/h ■■          ■
         ■            ■
  0 km/h ■■■■■■■■■■■■■■■■
         ■■■■■■■■■■■■■■■■■■■■■ tempo (s)

Nota: Curva suave = sem travamento
```

### Teste com ABS Ativo

```
200 km/h ■
         ■     ■■
150 km/h ■    ■  ■
         ■   ■    ■
100 km/h ■  ■      ■■■■■■■
         ■ ■
 50 km/h ■■
         ■
  0 km/h ■■■■■■■■■■■■■■■■
         ■■■■■■■■■■■■■■■■■■■■■ tempo (s)

Nota: Ondulações = pulsação ABS
```

---

## ■ Checklist de Validação

- [ ] Compilação sem erros
- [ ] Interface visual renderiza corretamente
- [ ] Controles responsivos (aceleração/freio)
- [ ] Cenário Manual funciona
- [ ] Cenário Emergency Brake completa
- [ ] Cenário High Speed completa
- [ ] Cenário Repeated Braking completa
- [ ] Histórico de velocidade atualiza
- [ ] Pressão das rodas varia
- [ ] ABS ativa quando necessário
- [ ] Nenhuma roda travada por >1s com ABS
- [ ] Sensor mostra ruído realístico
- [ ] Pausa (SPACE) funciona
- [ ] Reset (R) funciona

- [ ] Saída (Q/ESC) funciona

---

## ■ Troubleshooting

### "Cannot find module" ao compilar

```
# Solução:
cargo clean
cargo build --release
... truncated 34 lines for this snapshot in markdown volume ...
```

## File Deep Dive: TUTORIALS.md

Role: project support artifact

Line count: 671 | Non-empty: 496 | Symbol count: 64

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | ■ Tutoriais e Exemplos - Sistema de Freio Autônomo com ABS | namespace and compile-time linkage |
| 3 | h2 | ■ Sumário de Tutoriais | namespace and compile-time linkage |
| 12 | h2 | ■ Iniciante: Primeiro Contato | namespace and compile-time linkage |
| 14 | h3 | Objetivo | namespace and compile-time linkage |
| 17 | h3 | Tutorial Passo-a-Passo | namespace and compile-time linkage |
| 19 | h4 | Etapa 1: Iniciar a Simulação (1 minuto) | namespace and compile-time linkage |
| 37 | h4 | Etapa 2: Aceleração Básica (2 minutos) | namespace and compile-time linkage |
| 61 | h4 | Etapa 3: Freio Sem ABS (2 minutos) | namespace and compile-time linkage |
| 87 | h4 | Etapa 4: Ativação do ABS (3 minutos) | namespace and compile-time linkage |
| 118 | h4 | Etapa 5: Reset e Pausa (1 minuto) | namespace and compile-time linkage |
| 139 | h3 | ■ Checklist Iniciante | namespace and compile-time linkage |
| 151 | h2 | ■ Intermediário: Análise de Cenários | namespace and compile-time linkage |
| 153 | h3 | Objetivo | namespace and compile-time linkage |
| 158 | h3 | Teste 1: Modo Automático - Emergency Brake | namespace and compile-time linkage |
| 160 | h4 | Preparação | namespace and compile-time linkage |
| 164 | h4 | Timeline Esperada | namespace and compile-time linkage |
| 176 | h4 | Coleta de Dados | namespace and compile-time linkage |
| 192 | h4 | Análise de Resultados | namespace and compile-time linkage |
| 210 | h3 | Teste 2: Modo Manual - Frenagem Progressiva | namespace and compile-time linkage |
| 212 | h4 | Preparação | namespace and compile-time linkage |
| 217 | h4 | Frenagem em Estágios | namespace and compile-time linkage |
| 244 | h4 | Gráfico de Pulsação Esperado | namespace and compile-time linkage |
| 265 | h3 | Teste 3: Velocidade Sensor vs Real | namespace and compile-time linkage |
| 267 | h4 | Configuração | namespace and compile-time linkage |
| 271 | h4 | Coleta de Dados | namespace and compile-time linkage |
| 283 | h4 | Análise Estatística | namespace and compile-time linkage |
| 306 | h3 | ■ Checklist Intermediário | namespace and compile-time linkage |
| 317 | h2 | ■ Avançado: Interpretação de Dados | namespace and compile-time linkage |
| 319 | h3 | Objetivo | namespace and compile-time linkage |
| 324 | h3 | Análise 1: Dinâmica de Frenagem Comparativa | namespace and compile-time linkage |
| 326 | h4 | Experimento: Com vs Sem ABS | namespace and compile-time linkage |
| 356 | h3 | Análise 2: Eficácia do ABS por Velocidade | namespace and compile-time linkage |
| 358 | h4 | Teste: ABS em Diferentes Velocidades Iniciais | namespace and compile-time linkage |
| 376 | h4 | Interpretação | namespace and compile-time linkage |
| 393 | h3 | Análise 3: Padrão de Pulsação ABS | namespace and compile-time linkage |
| 395 | h4 | Observação Detalhada | namespace and compile-time linkage |
| 407 | h4 | Verificação de Simetria | namespace and compile-time linkage |
| 425 | h3 | Análise 4: Precisão do Sensor | namespace and compile-time linkage |
| 427 | h4 | Teste: Distribuição de Ruído | namespace and compile-time linkage |
| 451 | h3 | ■ Checklist Avançado | namespace and compile-time linkage |
| 462 | h2 | ■ Profissional: Estudo de Caso | namespace and compile-time linkage |
| 464 | h3 | Objetivo | namespace and compile-time linkage |
| 469 | h3 | Caso 1: Análise de Distância de Parada Segura | namespace and compile-time linkage |
| 471 | h4 | Cenário Real | namespace and compile-time linkage |
| 474 | h4 | Simulação | namespace and compile-time linkage |
| 488 | h4 | Análise de Segurança | namespace and compile-time linkage |
```

```
| 505 | h3 | Caso 2: Validação de Algoritmo ABS Customizado | namespace and compile-time linkage |
| 507 | h4 | Proposta: Teste de Algoritmo Alternativo | namespace and compile-time linkage |
| 529 | h4 | Comparação Experimental | namespace and compile-time linkage |
| 538 | h4 | Conclusão Técnica | namespace and compile-time linkage |
| 556 | h3 | Caso 3: Estudo de Robustez em Cenário Crítico | namespace and compile-time linkage |
| 558 | h4 | Teste: Múltiplas Frenagens Seguidas | namespace and compile-time linkage |
| 564 | h4 | Coleta de Dados | namespace and compile-time linkage |
| 582 | h3 | Caso 4: Análise de Falha Hipotética | namespace and compile-time linkage |
| 584 | h4 | Cenário: ABS Desativado | namespace and compile-time linkage |
| 600 | h4 | Importância em Produção | namespace and compile-time linkage |
| 616 | h3 | ■ Checklist Profissional | namespace and compile-time linkage |
| 627 | h2 | ■ Referências de Estudo | namespace and compile-time linkage |
| 629 | h3 | Documentação do Projeto | namespace and compile-time linkage |
| 634 | h3 | Recursos Externos Recomendados | namespace and compile-time linkage |
| 636 | h4 | ABS - Conceitos | namespace and compile-time linkage |
| 641 | h4 | Controle em Tempo Real | namespace and compile-time linkage |
| 645 | h4 | Simulação em Rust | namespace and compile-time linkage |
| 652 | h2 | ■ Conclusão | namespace and compile-time linkage |
```

## Content Snapshot

```
# ■ Tutoriais e Exemplos - Sistema de Freio Autônomo com ABS

## ■ Sumário de Tutoriais

1. [Iniciante: Primeiro Contato](#iniciante-primeiro-contato)
2. [Intermediário: Análise de Cenários](#intermediário-análise-de-cenários)
3. [Avançado: Interpretação de Dados](#avançado-interpretação-de-dados)
4. [Profissional: Estudo de Caso](#profissional-estudo-de-caso)

---

## ■ Iniciante: Primeiro Contato

### Objetivo
Familiarizar-se com a interface e entender funcionamento básico.

### Tutorial Passo-a-Passo

#### Etapa 1: Iniciar a Simulação (1 minuto)
```

cd AutoBreaking

cargo run

```
**O que você verá**:

■■■
```

Histórico: Gráfico sobe até 60 km/h

```
3. Espere 3 segundos sem pressionar nada
   - Pressione **↑** mais 2 vezes até ~100 km/h
   - Observe histórico preenchendo de baixo para cima

**Conceitos Aprendidos**:
- ✓ Controle de aceleração
- ✓ Leitura de painel
- ✓ Histórico de velocidade

---

#### Etapa 3: Freio Sem ABS (2 minutos)

1. Com velocidade em ~100 km/h, pressione **→** (freio)
   - Pressione **1x** para +10% freio

2. Observe:
```

Brake: [■■■■■■■■■■■■■■■■■■■■] 10.0%

Wheels:

FL ■ ROLLING | Vel: 100.0 km/h

FR ■ ROLLING | Vel: 100.0 km/h

RL ■ ROLLING | Vel: 100.0 km/h

RR ■ ROLLING | Vel: 100.0 km/h

```
3. Pressione **→** mais 2 vezes para 30% de freio
   - Velocidade começa a diminuir
   - Rodas ainda ■ ROLLING (não há travamento)

**Conceitos Aprendidos**:
- ✓ Controle de freio progressivo
- ✓ Estados das rodas (Rolling, Skidding, ABS)
- ✓ Sem travamento em frenagem leve

---

#### Etapa 4: Ativação do ABS (3 minutos)

1. Aumente freio para 50% (pressione **→** 2 mais vezes)

2. **Importante**: Observe mudança de estado:
```

Antes: Wheels = ■ ROLLING

Depois: Wheels = ■ ABS (ou ■ SKIDDING)

ABS Status: ■ ACTIVE

ABS Cycles: ↑↑↑ (aumenta)

```
3. Observe pulsação no gráfico de pressão:
```

FL [■■■■■■■■■■■] 60% ← Pulsando

FR [■■■■■■■■■■■] 30% ← Fase diferente

RL [■■■■■■■■■■■] 40%

RR [■■■■■■■■■■■] 20%

```
4. Espere até parada completa (~5-6 segundos)

**Conceitos Aprendidos**:
- ✓ Detecção de travamento (ΔV > 5 km/h)
```

```
- ✓ Ativação automática de ABS
- ✓ Pulsação a 8 Hz
- ✓ Pressões individuais por roda

---

#### Etapa 5: Reset e Pausa (1 minuto)

1. Pressione **R** para resetar
   - Tudo volta a zero
   - Histórico limpo

2. Pressione **↑** para acelerar de novo

3. Em algum momento, pressione **SPACE**
   - Simulação pausa
   - Velocidade congela
   - Mensagem "■ PAUSED"

4. Pressione **SPACE** novamente para retomar

**Conceitos Aprendidos**:
- ✓ Funções de controle (reset, pausa)
- ✓ Análise estática de um momento

---

### ■ Checklist Iniciante

- [ ] Compilou e rodou sem erros
- [ ] Entende barra de velocidade
- [ ] Conseguiu acelerar até 100 km/h
- [ ] Conseguiu frear completamente
- [ ] Viu ABS ser ativado
- [ ] Observou mudança de estado das rodas
- [ ] Usou pausa e reset

---

## ■ Intermediário: Análise de Cenários

### Objetivo
Entender diferentes modos de simulação e analisar dados.

---

### Teste 1: Modo Automático - Emergency Brake

#### Preparação
1. Pressione **2** para ativar "Emergency Brake"
2. Sistema fará tudo automaticamente

#### Timeline Esperada
```

t=0s Começa aceleração

t=0.5s Velocidade ~25 km/h

t=1.0s Velocidade ~50 km/h

t=1.5s Velocidade ~75 km/h

t=2.0s Velocidade ~100 km/h

t=3.0s ← Freio máximo aplicado!

t=3.2s ABS ativado (ΔV > 5)

t=5.5s Parada completa

```
#### Coleta de Dados
1. **Velocidade**:
```

```
   - Inicial: 0 km/h
   - Máxima: ~100-110 km/h (sensor pode ler diferente)
   - Final: 0 km/h
   - Tempo total: ~5.5s

2. **Ciclos ABS**:
   - Esperado: 8-12 ciclos (8 Hz × 1-1.5s)
   - Padrão: Regular e consistente

3. **Estados das Rodas**:
   - Esperado: ■ ABS constantemente durante frenagem
   - Nunca ■ SKIDDING (ABS previne)

#### Análise de Resultados
```

Pergunta: Por que ABS não foi ativado imediatamente?

Resposta: Condições para ABS:

1. ΔV > 5 km/h (velocidade roda vs veículo)

2. Brake > 0.3 (30% de freio)

3. Velocidade veículo > 5 km/h

Em baixas velocidades (<20 km/h), travamento

é detectado mas não é crítico.

Portanto: ABS atua principalmente em

desaceleração de 100 para 0 km/h.

```
---

### Teste 2: Modo Manual - Frenagem Progressiva

#### Preparação
1. Pressione **1** para "Manual"
2. Acelere a 80 km/h (↑ 5 vezes)
3. Agora vamos frear lentamente

#### Frenagem em Estágios

**Estágio 1: 20% de freio**
```

... truncated 451 lines for this snapshot in markdown volume ...

```
## File Deep Dive: allowlist.example.json
Role: project support artifact
Line count: 16 | Non-empty: 16 | Symbol count: 0

### Symbol Index
No symbols extracted for this file type.

### Content Snapshot
```

[

{

"type": "can",

"id": 419385573,

"mask": "0x1FFFFFFF",

"allowed_bytes": [[2, 2]],

"max_rate_per_sec": 1,

"description": "Allowed two-byte write at payload offset 2"

},

{

"type": "serial",

"pattern_hex": "AA55..",

"max_rate_per_sec": 1,

"description": "Allowed bootloader unlock pattern"

}

]

```
## File Deep Dive: benches/solver_bench.rs
Role: project support artifact
Line count: 32 | Non-empty: 29 | Symbol count: 1

### Function Call Hotspots
| Function | Approx Calls In File |
|---|---:|
| key_advance | 6 |
| new | 2 |
| tick | 2 |
| bench_tick | 1 |

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 3 | fn | bench_tick | API and behavior coupling to callers/implementers |

### Line by Line Change Impact
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use criterion::{criterion_group, criterion_main, Criterion}; | import wiring; changing path/name can break compi
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | fn bench_tick(c: &mut Criterion) { | function signature/body; changes affect API behavior and control-flow. |
| 4 |     c.bench_function("tick_1k_steps", \|b\| { | support logic; impact depends on neighboring block and data flow
| 5 |         b.iter(\|\| { | support logic; impact depends on neighboring block and data flow. |
| 6 |             let mut sim = auto_breaking::HeavyMachinery::new(); | support logic; impact depends on neighboring b
| 7 |             sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 8 |             sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 9 |             sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 10 |             for _ in 0..1000 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 11 |                 sim.throttle_pct = 40.0; | support logic; impact depends on neighboring block and data flow. |
| 12 |                 sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 13 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 14 |         }) | support logic; impact depends on neighboring block and data flow. |
| 15 |     }); | support logic; impact depends on neighboring block and data flow. |
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 |     c.bench_function("tick_10k_steps", \|b\| { | support logic; impact depends on neighboring block and data fl
| 18 |         b.iter(\|\| { | support logic; impact depends on neighboring block and data flow. |
| 19 |             let mut sim = auto_breaking::HeavyMachinery::new(); | support logic; impact depends on neighboring
| 20 |             sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 21 |             sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 22 |             sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 23 |             for _ in 0..10_000 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 24 |                 sim.throttle_pct = 35.0; | support logic; impact depends on neighboring block and data flow. |
| 25 |                 sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 26 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 |         }) | support logic; impact depends on neighboring block and data flow. |
| 28 |     }); | support logic; impact depends on neighboring block and data flow. |
| 29 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 30 | (blank) | blank separator; no runtime behavior. |
| 31 | criterion_group!(benches, bench_tick); | support logic; impact depends on neighboring block and data flow. |
| 32 | criterion_main!(benches); | support logic; impact depends on neighboring block and data flow. |

## File Deep Dive: docs/AI_AGENT_RUNBOOK.md
```

```
Role: human-facing architecture, protocols, runbooks, and design records
Line count: 29 | Non-empty: 25 | Symbol count: 5

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | AI Agent Refactor Runbook | namespace and compile-time linkage |
| 3 | h2 | Branch strategy | namespace and compile-time linkage |
| 7 | h2 | Agent cycle | namespace and compile-time linkage |
| 13 | h2 | Prompt template | namespace and compile-time linkage |
| 21 | h2 | PR acceptance checklist | namespace and compile-time linkage |

### Content Snapshot
```

# AI Agent Refactor Runbook

## Branch strategy

- Base branch: `feat/auto-refactor-runner`
- Agent branch: `feat/ai-auto-refactor`

## Agent cycle

1. Select 5-10 files max.
2. Ask for 1-3 atomic changes per cycle.
3. Require tests and changelog updates.
4. Validate locally: `cargo fmt -- --check`, `cargo clippy -- -D warnings`, `cargo test`.

## Prompt template

"You are a senior Rust engineer for embedded simulation systems. Analyze up to 10 files and produce:

1) max 5 bullets of findings,

2) unified diff in 1-2 logical commits,

3) tests (unit or property-based),

4) exact verification commands.

Do not break public API without compatibility shim + migration notes."

## PR acceptance checklist

- [ ] Changelog updated
- [ ] Unit tests added/updated
- [ ] Integration tests added/updated when cross-module
- [ ] `cargo fmt -- --check` passes
- [ ] `cargo clippy -- -D warnings` passes
- [ ] `cargo test` passes
- [ ] Benchmarks included if performance-sensitive
- [ ] Feature flag used for risky rollout

```
## File Deep Dive: docs/CAT_COMM_TEMPLATE_PROTOCOL.md
```

```
Role: human-facing architecture, protocols, runbooks, and design records
Line count: 189 | Non-empty: 141 | Symbol count: 13

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Cat Comm Template Bridge Protocol (Upload-Ready) | namespace and compile-time linkage |
| 7 | h2 | 1. Startup modes | namespace and compile-time linkage |
| 15 | h2 | 2. Transport contract | namespace and compile-time linkage |
| 23 | h2 | 3. Request schema | namespace and compile-time linkage |
| 27 | h3 | init | namespace and compile-time linkage |
| 40 | h3 | ping | namespace and compile-time linkage |
| 46 | h3 | read | namespace and compile-time linkage |
| 55 | h3 | try_read | namespace and compile-time linkage |
| 61 | h3 | send | namespace and compile-time linkage |
| 93 | h3 | close | namespace and compile-time linkage |
| 99 | h2 | 4. Response schema | namespace and compile-time linkage |
| 170 | h2 | 5. Upload layout options | namespace and compile-time linkage |
| 182 | h2 | 6. Safety expectations | namespace and compile-time linkage |

### Content Snapshot
```

# Cat Comm Template Bridge Protocol (Upload-Ready)

This document defines the bridge contract expected by the Windows Cat Comm adapter in [src/io/vendor_cat_comm.rs](src/io/vendor_cat_comm.rs).

The repository includes a functional reference bridge implementation at [src/bin/cat_comm_bridge.rs](src/bin/cat_comm_bridge.rs) and a staging helper script at [scripts/build_cat_comm_template_bridge.ps1](scripts/build_cat_comm_template_bridge.ps1).

## 1. Startup modes

The bridge executable must support:

1. `--mode=stdio-jsonl`

The host process launches the bridge and communicates with JSON lines over stdin/stdout.

## 2. Transport contract

- One JSON object per line.
- UTF-8 encoded.
- Requests are sent by the host.
- Responses are returned by the bridge.
- Every request requires one response line.

## 3. Request schema

`cmd` is required.

### init

```
{
```

```
  "cmd": "init",
  "dry_run": true,
  "enable_write": false,
  "can_interface": "can0",
  "serial_port": "COM5",
  "template_dir": "C:/path/to/template"
}
```

### ping

```
{ "cmd": "ping" }
```

### read

```
{
  "cmd": "read",
  "timeout_ms": 1000
}
```

### try_read

```
{ "cmd": "try_read" }
```

### send

```
{
  "cmd": "send",
  "frame": {
    "Can": {
      "id": 418381568,
      "dlc": 8,
      "data": [1,2,3,4,5,6,7,8],
      "len": 8,
      "timestamp_ms": 1723520000000
    }
  }
}
```

Serial frame variant:

```
{
  "cmd": "send",
  "frame": {
    "Serial": {
      "bytes": [62,0],
      "protocol_hint": "uds-raw",
      "timestamp_ms": 1723520000000
    }
  }
}
```

### close

```
{ "cmd": "close" }
```

## 4. Response schema

Base fields:

```
{
  "ok": true,
  "frame": null,
  "error": null,
  "kind": null
```

}

Optional capability negotiation fields (recommended):

{

"ok": true,

"protocol_version": "1.0",

"capabilities": ["can", "serial", "write", "uds_flash"]

}

Optional transport diagnostics fields (for flash telemetry parity):

{

"ok": true,

"diag": {

"wait_frame_count_delta": 1,

"sequence_error_count_delta": 0,

"flowcontrol_timeout_count_delta": 0,

"fc_blocksize": 8,

"fc_stmin_ms": 5

}

}

When provided, host appends these deltas to `log_dir/cat_comm_bridge_diag.jsonl` and merges them into live flash trans

On `read`/`try_read`, include `frame` when data exists.

On failure:

```
{
  "ok": false,
  "error": "human-readable detail",
  "kind": "timeout"
}
```

Supported `kind` values mapped by host:

1. `timeout`
2. `rate_limited`
3. `bus_off`
4. `transceiver`
5. `permission`
6. `write_blocked`
7. `parse`

Any other value maps to generic `Unknown` host error.

Compatibility notes:

1. `protocol_version` is optional for backward compatibility.
2. If present, host currently accepts major version `1`.
3. `capabilities` is optional.
4. If `capabilities` is provided, host enforces strict checks for requested transport and write mode.

## 5. Upload layout options

Option A: explicit executable path

1. Put bridge executable anywhere.
2. Launch AutoBreaking with `--vendor-name=cat_comm --vendor-bridge-exe=<full_path_to_exe>`.

Option B: template directory convention

1. Put executable as `cat_comm_bridge.exe` inside your uploaded template folder.
2. Launch with `--vendor-name=cat_comm --vendor-template-dir=<folder>`.

## 6. Safety expectations

The bridge must fail closed:

1. Never claim success if transport write/read failed.
2. Return `ok=false` + `error` + `kind` on faults.
3. Respect `enable_write` from init request.
4. Preserve deterministic timeout behavior for `read`.

```
## File Deep Dive: docs/DECISIONS.md
Role: human-facing architecture, protocols, runbooks, and design records
Line count: 17 | Non-empty: 13 | Symbol count: 5

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Architecture Decisions | namespace and compile-time linkage |
| 3 | h2 | ADR-001: Incremental refactor with API preservation | namespace and compile-time linkage |
| 7 | h2 | ADR-002: Feature flags for high-risk changes | namespace and compile-time linkage |
| 11 | h2 | ADR-003: Test matrix-first workflow | namespace and compile-time linkage |
| 15 | h2 | ADR-004: Structured telemetry baseline | namespace and compile-time linkage |

### Content Snapshot
```

# Architecture Decisions

## ADR-001: Incremental refactor with API preservation

- We introduce internal traits and helper modules before changing public APIs.
- Reason: reduce regression risk and keep UI and integration tests stable.

## ADR-002: Feature flags for high-risk changes

- Flags: `advanced_observability`, `new_integrator`.
- Reason: safe rollout/rollback without large reverts.

## ADR-003: Test matrix-first workflow

- Every refactor commit must include unit or integration coverage.
- Nightly Monte Carlo and fuzz smoke run in CI to catch long-tail failures.

## ADR-004: Structured telemetry baseline

- Logs use JSON event format with correlation IDs.
- Metrics baseline includes step duration, steps completed, errors, replay failures.

```
## File Deep Dive: docs/ECM-Data.md
Role: human-facing architecture, protocols, runbooks, and design records
Line count: 269 | Non-empty: 183 | Symbol count: 16

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | ECM-Data Production Runbook (Windows-first) | namespace and compile-time linkage |
| 3 | h2 | 0. Executive summary | namespace and compile-time linkage |
| 7 | h2 | 1. Goals and non-goals | namespace and compile-time linkage |
| 21 | h2 | 2. Hardware adapter model | namespace and compile-time linkage |
| 47 | h2 | 3. Safety controls | namespace and compile-time linkage |
| 62 | h2 | 4. Allowlist schema and enforcement | namespace and compile-time linkage |
| 81 | h2 | 5. Rate limiter sizing | namespace and compile-time linkage |
| 99 | h2 | 6. Log and storage sizing | namespace and compile-time linkage |
| 116 | h2 | 7. Metrics model | namespace and compile-time linkage |
| 135 | h2 | 8. Failure modes and mitigations | namespace and compile-time linkage |
| 150 | h2 | 9. Windows Cat Comm runbook | namespace and compile-time linkage |
| 176 | h2 | 10. CI and test matrix | namespace and compile-time linkage |
| 197 | h2 | 11. Security checklist | namespace and compile-time linkage |
| 206 | h2 | 12. Upgrade roadmap | namespace and compile-time linkage |
| 214 | h2 | 13. One-shot 12-phase execution | namespace and compile-time linkage |
| 252 | h2 | 14. Linux can-utils interoperability validation | namespace and compile-time linkage |

### Content Snapshot
```

# ECM-Data Production Runbook (Windows-first)

## 0. Executive summary

AutoBreaking includes a production-grade ECM live integration layer with safe-by-default writes, operator-gated actions, audit logging, replay support, and test adapters. Windows operation is first-class through a Cat Comm integration seam and COM serial fallback.

## 1. Goals and non-goals

Goals:

1. Reliable asynchronous read and controlled write policy for live ECM communications.
2. Safe-by-default write posture with deny-unless-explicit gates.
3. Full audit trail via JSONL logs and startup policy evidence.
4. CI and testability without hardware using mock adapters.

Non-goals (current phase):

1. Bundling proprietary vendor SDK binaries in repository.
2. Full proprietary parser coverage without approved captures/specs.

## 2. Hardware adapter model

The adapter contract is centralized in [src/io/hw.rs](src/io/hw.rs).

Implementations:

1. Serial adapter for COM links: [src/io/serial_adapter.rs](src/io/serial_adapter.rs)
2. SocketCAN adapter for Linux: [src/io/socketcan_adapter.rs](src/io/socketcan_adapter.rs)
3. Cat Comm template (Windows): [src/io/vendor_cat_comm.rs](src/io/vendor_cat_comm.rs)
4. Cat Comm bridge protocol: [docs/CAT_COMM_TEMPLATE_PROTOCOL.md](docs/CAT_COMM_TEMPLATE_PROTOCOL.md)
5. Reference bridge executable source: [src/bin/cat_comm_bridge.rs](src/bin/cat_comm_bridge.rs)
6. Mock adapter for tests: [src/io/mock.rs](src/io/mock.rs)

Selection order:

1. vendor_name=cat_comm (Windows + feature vendor-windows)
2. can-if
3. serial-port

Cat Comm bridge configuration flags:

1. --vendor-name=cat_comm
2. --vendor-bridge-exe=<path_to_cat_comm_bridge.exe>
3. --vendor-template-dir=<folder_containing_cat_comm_bridge.exe>
4. --vendor-bridge-timeout-ms=<timeout_ms>

## 3. Safety controls

Write must satisfy all controls:

1. enable-write true
2. allowlist path present and valid
3. noninteractive-approved true
4. dry-run false

Code references:

1. [src/io/hw.rs](src/io/hw.rs)

2. [src/io/allowlist.rs](src/io/allowlist.rs)
3. [src/io/rate_limiter.rs](src/io/rate_limiter.rs)

## 4. Allowlist schema and enforcement

Example file:

1. [allowlist.example.json](allowlist.example.json)

Current rule capabilities:

1. CAN id + optional mask
2. payload windows with allowed_bytes
3. serial hex patterns with wildcard byte pairs
4. optional per-rule max_rate_per_sec

Recommended hardening:

1. signed allowlist plus detached signature
2. validation at startup/reload
3. operator identity trace in audit stream

## 5. Rate limiter sizing

Two-level limiter model:

1. Global token bucket
2. Per-CAN-ID token bucket

Recommended conservative profile:

1. global rate: 100 tx/s
2. global burst: 300 tokens
3. per-id rate: 1 tx/s
4. per-id burst: 5 tokens

Token bucket equation:

$$tokens(t) = min(C, tokens(t_0) + r (t - t_0))$$

## 6. Log and storage sizing

Live outputs:

1. startup audit gzip jsonl

2. ecm_rx jsonl
3. ecm_snapshot jsonl
4. exported CSV snapshots from ECM-Live tab

Sizing assumptions:

1. CAN RX line ~120 bytes
2. At 1000 frames/s:
1. per day raw: about 10.4 GB
2. 30-day raw: about 312 GB
3. gzip reduction commonly 4x to 8x

## 7. Metrics model

Current in-process counters:

1. [src/io/metrics.rs](src/io/metrics.rs)

Recommended Prometheus extension:

1. hw_rx_frames_total by transport
2. hw_tx_frames_total by transport and allowed verdict
3. hw_read_errors_total and hw_write_errors_total
4. hw_rate_limited_total
5. hw_last_rx_timestamp

Cardinality guidance:

1. Avoid frame-id labels in primary timeseries.
2. Use aggregated source labels only.

## 8. Failure modes and mitigations

1. COM permission denied:

Validate driver install and process privileges.

2. Device disconnect:

Use reconnect backoff and operator alerts.

3. Bus-off or transceiver error:

Stop writes and require controlled recovery sequence.

4. Parser corruption:

Use resync boundaries and retain raw forensic logs.

5. Allowlist regression:

Require staged rollout: dry-run first, then controlled write.

6. Vendor SDK crash:

Run vendor SDK in bridge subprocess with watchdog restart.

## 9. Windows Cat Comm runbook

Preconditions:

1. bench isolation and operator authorization
2. emergency kill-switch available
3. validated COM/vendor driver path

Checks:

```
Get-WmiObject Win32_SerialPort | Select-Object DeviceID, Caption
pnputil /enum-drivers | findstr /i "Cat"
```

Safe flow:

1. launch with dry-run
2. for Cat Comm template upload, provide one of:
1. --vendor-bridge-exe=<full_path>
2. --vendor-template-dir=<template_folder>
2. Detect then Connect then Retrieve Data in ECM-Live tab
3. verify live snapshot updates and jsonl outputs
4. export CSV and review trends
5. only then move to dry-run=false with explicit approvals

## 10. CI and test matrix

Workflow:

1. [/.github/workflows/integration-mock.yml](.github/workflows/integration-mock.yml)

Coverage:

1. formatting
2. clippy target tests
3. io mock integration suite

Local command set:

```
cargo fmt
cargo clippy --test io_mock
cargo test --test io_mock
cargo check
```

## 11. Security checklist

1. default deny writes
2. least privilege runtime account
3. protected allowlist storage and ACLs
4. append-only audit logs
5. metrics endpoint restricted to trusted network path
6. no secrets in repo or plain-text configs

## 12. Upgrade roadmap

1. implemented: Cat Comm bridge protocol + reference executable source
2. open: add signed allowlist verification
3. open: add Prometheus exporter endpoint
4. open: parser extension with real captures
5. implemented: Windows template E2E runner script [scripts/run_windows_template_e2e.ps1](scripts/run_windows_template_e2e.ps1)

## 13. One-shot 12-phase execution

Run all production phases in a single non-interactive pass:

```
cargo run --bin simulator_cli -- --run-production-phases --hw-mode=live --vendor-name=cat_comm --vendor-template-dir="
```

... truncated 49 lines for this snapshot in markdown volume ...

```
## File Deep Dive: docs/FULL_BOOK_ULTIMATE.md
Role: human-facing architecture, protocols, runbooks, and design records
Line count: 43179 | Non-empty: 42217 | Symbol count: 645

### Symbol Index
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | AutoBreaking Ultimate Full Book | namespace and compile-time linkage |
| 5 | h2 | Meta | namespace and compile-time linkage |
| 11 | h2 | Architecture Narrative | namespace and compile-time linkage |
| 18 | h2 | Module Coupling Graph | namespace and compile-time linkage |
| 44 | h2 | Dependency Tree Snapshot | namespace and compile-time linkage |
| 438 | h2 | Working Tree Status | namespace and compile-time linkage |
| 467 | h2 | Master File Catalog | namespace and compile-time linkage |
| 576 | h2 | File Deep Dive: .github/pull_request_template.md | namespace and compile-time linkage |
| 580 | h3 | Symbol Index | namespace and compile-time linkage |
| 589 | h3 | Content Snapshot | namespace and compile-time linkage |
| 591 | h2 | Problem | namespace and compile-time linkage |
| 594 | h2 | Changes | namespace and compile-time linkage |
| 600 | h2 | How to test | namespace and compile-time linkage |
| 607 | h2 | Artifacts | namespace and compile-time linkage |
| 611 | h2 | Acceptance checklist | namespace and compile-time linkage |
| 620 | h2 | File Deep Dive: .github/workflows/ci.yml | namespace and compile-time linkage |
| 624 | h3 | Symbol Index | namespace and compile-time linkage |
| 627 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 652 | h2 | File Deep Dive: .github/workflows/fuzz-smoke.yml | namespace and compile-time linkage |
| 656 | h3 | Symbol Index | namespace and compile-time linkage |
| 659 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 681 | h2 | File Deep Dive: .github/workflows/integration-mock.yml | namespace and compile-time linkage |
| 685 | h3 | Symbol Index | namespace and compile-time linkage |
```

| 688 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 732 | h2 | File Deep Dive: .github/workflows/nightly-montecarlo.yml | namespace and compile-time linkage |
| 736 | h3 | Symbol Index | namespace and compile-time linkage |
| 739 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 768 | h2 | File Deep Dive: AUDIT_FULL_2026-08-12.md | namespace and compile-time linkage |
| 772 | h3 | Symbol Index | namespace and compile-time linkage |
| 788 | h3 | Content Snapshot | namespace and compile-time linkage |
| 790 | h1 | Full Technical Audit - AutoBreaking | namespace and compile-time linkage |
| 796 | h2 | 1. What was executed | namespace and compile-time linkage |
| 808 | h2 | 2. Execution evidence summary | namespace and compile-time linkage |
| 827 | h2 | 3. Severity summary | namespace and compile-time linkage |
| 840 | h2 | 4. Confirmed findings (highest priority) | namespace and compile-time linkage |
| 842 | h3 | High | namespace and compile-time linkage |
| 858 | h3 | Medium | namespace and compile-time linkage |
| 892 | h2 | 5. Agent-flagged findings requiring targeted confirmation | namespace and compile-time linkage |
| 901 | h2 | 6. Module-by-module status | namespace and compile-time linkage |
| 945 | h2 | 7. API and integration audit conclusions | namespace and compile-time linkage |
| 951 | h2 | 8. Recommended remediation roadmap | namespace and compile-time linkage |
| 971 | h2 | 9. Final audit verdict | namespace and compile-time linkage |
| 978 | h2 | File Deep Dive: CHANGELOG.md | namespace and compile-time linkage |
| 982 | h3 | Symbol Index | namespace and compile-time linkage |
| 990 | h3 | Content Snapshot | namespace and compile-time linkage |
| 992 | h1 | Changelog | namespace and compile-time linkage |
| 994 | h2 | [Unreleased] | namespace and compile-time linkage |
| 995 | h3 | Added | namespace and compile-time linkage |
| 1005 | h3 | Changed | namespace and compile-time linkage |
| 1010 | h2 | File Deep Dive: CODE_EXAMPLES.md | namespace and compile-time linkage |
| 1014 | h3 | Symbol Index | namespace and compile-time linkage |
| 1061 | h3 | Content Snapshot | namespace and compile-time linkage |
| 1063 | h1 | ■ Exemplos de Código - Sistema de Freio com ABS | namespace and compile-time linkage |
| 1069 | h2 | 1. Exemplo: Modificar Frequência ABS | namespace and compile-time linkage |
| 1071 | h3 | Localização | namespace and compile-time linkage |
| 1074 | h3 | Código Original | namespace and compile-time linkage |
| 1082 | h3 | Modificação: Aumentar para 10 Hz | namespace and compile-time linkage |
| 1098 | h3 | Recompilar e Testar | namespace and compile-time linkage |
| 1102 | h1 | Selecione Emergency Brake (tecla 2) | namespace and compile-time linkage |
| 1103 | h1 | Observe ciclos ABS acontecerem ~25% mais rápido | namespace and compile-time linkage |
| 1108 | h2 | 2. Exemplo: Ajustar Limiar de Travamento | namespace and compile-time linkage |
| 1110 | h3 | Localização | namespace and compile-time linkage |
| 1113 | h3 | Código Original | namespace and compile-time linkage |
| 1121 | h3 | Modificação: Mais Sensível (4 km/h) | namespace and compile-time linkage |
| 1129 | h3 | Impacto | namespace and compile-time linkage |
| 1136 | h2 | 3. Exemplo: Mudar Pressão de Liberação ABS | namespace and compile-time linkage |
| 1138 | h3 | Localização | namespace and compile-time linkage |
| 1141 | h3 | Código Original | namespace and compile-time linkage |
| 1150 | h3 | Modificação: Mais Agressivo | namespace and compile-time linkage |
| 1159 | h3 | Modificação: Mais Conservador | namespace and compile-time linkage |
| 1170 | h2 | 4. Exemplo: Adicionar Novo Cenário | namespace and compile-time linkage |
| 1172 | h3 | Passo 1: Adicionar Variante de Enum | namespace and compile-time linkage |
| 1186 | h3 | Passo 2: Implementar Display | namespace and compile-time linkage |
| 1200 | h3 | Passo 3: Implementar Lógica | namespace and compile-time linkage |
| 1220 | h3 | Passo 4: Adicionar Tecla de Ativação | namespace and compile-time linkage |
| 1231 | h3 | Passo 5: Atualizar Controles (comentário) | namespace and compile-time linkage |
| 1241 | h2 | 5. Exemplo: Implementar Função de Análise | namespace and compile-time linkage |
| 1243 | h3 | Novo Módulo: metrics.rs | namespace and compile-time linkage |
| 1286 | h2 | File Deep Dive: Cargo.toml | namespace and compile-time linkage |
| 1290 | h3 | Symbol Index | namespace and compile-time linkage |
| 1293 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 1337 | h2 | File Deep Dive: J1939_CAN_MANUAL.md | namespace and compile-time linkage |
| 1341 | h3 | Symbol Index | namespace and compile-time linkage |
| 1456 | h3 | Content Snapshot | namespace and compile-time linkage |
| 1458 | h1 | Manual Completo: CAN Bus e J1939 | namespace and compile-time linkage |
| 1459 | h2 | Guia Definitivo para Veículos Pesados e Maquinário Agrícola | namespace and compile-time linkage |
| 1463 | h1 | PARTE 1: CAN BUS — FUNDAMENTOS COMPLETOS | namespace and compile-time linkage |
| 1465 | h2 | 1.1 O que é o CAN Bus | namespace and compile-time linkage |
| 1469 | h3 | Por que CAN e não Ethernet ou RS-485? | namespace and compile-time linkage |
| 1482 | h2 | 1.2 Camada Física (Physical Layer) — ISO 11898-2 | namespace and compile-time linkage |
| 1484 | h3 | Diferencial e Topologia | namespace and compile-time linkage |
| 1501 | h3 | Níveis de Tensão | namespace and compile-time linkage |
| 1517 | h3 | Terminadores | namespace and compile-time linkage |
| 1527 | h3 | Velocidades e Comprimentos Máximos | namespace and compile-time linkage |

| 1540 | h2 | 1.3 Formato do Frame CAN 2.0 | namespace and compile-time linkage |
| 1542 | h3 | Frame de Dados (Data Frame) | namespace and compile-time linkage |
| 1571 | h3 | Bits por Frame (CAN 2.0B, 8 bytes dados, sem bit stuffing) | namespace and compile-time linkage |
| 1582 | h2 | 1.4 Arbitração — Como Múltiplos Nós Compartilham o Barramento | namespace and compile-time linkage |
| 1612 | h2 | 1.5 Tipos de Frames CAN | namespace and compile-time linkage |
| 1614 | h3 | Data Frame | namespace and compile-time linkage |
| 1617 | h3 | Remote Frame (RTR=1) | namespace and compile-time linkage |
| 1620 | h3 | Error Frame | namespace and compile-time linkage |
| 1627 | h3 | Overload Frame | namespace and compile-time linkage |
| 1632 | h2 | 1.6 Detecção e Confinamento de Erros | namespace and compile-time linkage |
| 1634 | h3 | Tipos de Erro Detectados | namespace and compile-time linkage |
| 1644 | h3 | Contadores de Erro (ISO 11898-1) | namespace and compile-time linkage |
| 1657 | h3 | Estados do Nó (Error Confinement) | namespace and compile-time linkage |
| 1677 | h2 | 1.7 Bit Stuffing | namespace and compile-time linkage |
| 1681 | h2 | File Deep Dive: PROJECT_SUMMARY.md | namespace and compile-time linkage |
| 1685 | h3 | Symbol Index | namespace and compile-time linkage |
| 1742 | h3 | Content Snapshot | namespace and compile-time linkage |
| 1744 | h1 | ■ Sumário Completo do Projeto - Sistema de Freio Autônomo com ABS | namespace and compile-time linkage |
| 1746 | h2 | ■ O que você tem agora | namespace and compile-time linkage |
| 1752 | h2 | ■ Estrutura de Arquivos | namespace and compile-time linkage |
| 1774 | h2 | ■ Documentação Incluída | namespace and compile-time linkage |
| 1776 | h3 | 1. **README.md** (este arquivo) | namespace and compile-time linkage |
| 1784 | h3 | 2. **TESTING.md** | namespace and compile-time linkage |
| 1791 | h3 | 3. **ANALYSIS.md** | namespace and compile-time linkage |
| 1800 | h3 | 4. **TUTORIALS.md** | namespace and compile-time linkage |
| 1807 | h3 | 5. **ARCHITECTURE.md** | namespace and compile-time linkage |
| 1818 | h2 | ■ Recursos Principais | namespace and compile-time linkage |
| 1820 | h3 | Sistema de Simulação (lib.rs) | namespace and compile-time linkage |
| 1836 | h3 | Interface Visual (main.rs) | namespace and compile-time linkage |
| 1850 | h2 | ■ Como Começar | namespace and compile-time linkage |
| 1852 | h3 | 1. Compilação Inicial | namespace and compile-time linkage |
| 1859 | h3 | 2. Execução | namespace and compile-time linkage |
| 1865 | h3 | 3. Testar Cenários | namespace and compile-time linkage |
| 1874 | h2 | ■ Características Técnicas | namespace and compile-time linkage |
| 1876 | h3 | Performance | namespace and compile-time linkage |
| 1882 | h3 | Física | namespace and compile-time linkage |
| 1888 | h3 | Fidelidade | namespace and compile-time linkage |
| 1896 | h2 | ■ Dados Exportáveis | namespace and compile-time linkage |
| 1912 | h2 | ■ Casos de Uso | namespace and compile-time linkage |
| 1914 | h3 | Educacional | namespace and compile-time linkage |
| 1920 | h3 | Pesquisa | namespace and compile-time linkage |
| 1926 | h3 | Protótipo | namespace and compile-time linkage |
| 1934 | h2 | ■ Tecnologias Utilizadas | namespace and compile-time linkage |
| 1936 | h3 | Linguagem | namespace and compile-time linkage |
| 1939 | h3 | Dependências | namespace and compile-time linkage |
| 1945 | h3 | Padrões de Design | namespace and compile-time linkage |
| 1953 | h2 | ■ Validação e Testes | namespace and compile-time linkage |
| 1955 | h3 | Testes Inclusos | namespace and compile-time linkage |
| 1967 | h2 | File Deep Dive: README.md | namespace and compile-time linkage |
| 1971 | h3 | Symbol Index | namespace and compile-time linkage |
| 2003 | h3 | Content Snapshot | namespace and compile-time linkage |
| 2005 | h1 | SimuBench | namespace and compile-time linkage |
| 2016 | h2 | Visao Rapida | namespace and compile-time linkage |
| 2026 | h2 | Comece Aqui | namespace and compile-time linkage |
| 2028 | h3 | 1. Rodar em simulacao | namespace and compile-time linkage |
| 2034 | h3 | 2. Rodar o binario principal explicitamente | namespace and compile-time linkage |
| 2040 | h3 | 3. Rodar o runner headless | namespace and compile-time linkage |
| 2046 | h3 | 4. Validar qualidade | namespace and compile-time linkage |
| 2054 | h2 | O Que Voce Vai Ver Na UI | namespace and compile-time linkage |
| 2058 | h3 | Operacao | namespace and compile-time linkage |
| 2069 | h3 | Diagnostico | namespace and compile-time linkage |
| 2080 | h2 | Fluxo Recomendado | namespace and compile-time linkage |
| 2092 | h2 | Principais Capacidades | namespace and compile-time linkage |
| 2094 | h3 | Simulacao multi-ECU | namespace and compile-time linkage |
| 2102 | h3 | CAN, J1939 e UDS | namespace and compile-time linkage |
| 2117 | h3 | Sensores e autonomia | namespace and compile-time linkage |
| 2136 | h3 | Leak Physics Lab | namespace and compile-time linkage |
| 2149 | h2 | Leak Lab Em 30 Segundos | namespace and compile-time linkage |
| 2151 | h3 | O que ele faz | namespace and compile-time linkage |
| 2159 | h3 | Fluxo rapido | namespace and compile-time linkage |
| 2167 | h3 | CSV esperado para calibracao | namespace and compile-time linkage |

| 2185 | h2 | Modo Live e Seguranca | namespace and compile-time linkage |
| 2189 | h3 | Flags importantes | namespace and compile-time linkage |
| 2206 | h3 | Gates de escrita fisica | namespace and compile-time linkage |
| 2218 | h2 | Binarios | namespace and compile-time linkage |
| 2228 | h2 | File Deep Dive: TESTING.md | namespace and compile-time linkage |
| 2232 | h3 | Symbol Index | namespace and compile-time linkage |
| 2268 | h3 | Content Snapshot | namespace and compile-time linkage |
| 2270 | h1 | ■ Guia de Teste Rápido - Sistema Autônomo de Freio com ABS | namespace and compile-time linkage |
| 2272 | h2 | Verificação Inicial de Compilação | namespace and compile-time linkage |
| 2282 | h2 | ■ Cenas de Teste Recomendadas | namespace and compile-time linkage |
| 2284 | h3 | Teste 1: Freio Manual Progressivo (3-5 minutos) | namespace and compile-time linkage |
| 2303 | h3 | Teste 2: Freio de Emergência (2-3 minutos) | namespace and compile-time linkage |
| 2320 | h3 | Teste 3: Alta Velocidade (3-4 minutos) | namespace and compile-time linkage |
| 2335 | h3 | Teste 4: Frenagens Repetidas (4-5 minutos) | namespace and compile-time linkage |
| 2352 | h2 | ■ Métricas para Análise | namespace and compile-time linkage |
| 2354 | h3 | Velocidade | namespace and compile-time linkage |
| 2359 | h3 | Estado das Rodas | namespace and compile-time linkage |
| 2366 | h3 | Pressão de Freio | namespace and compile-time linkage |
| 2371 | h3 | Ciclos ABS | namespace and compile-time linkage |
| 2378 | h2 | ■ Observações de Engenharia | namespace and compile-time linkage |
| 2380 | h3 | Comportamento Normal ■ | namespace and compile-time linkage |
| 2389 | h3 | Comportamento Anômalo ■■ | namespace and compile-time linkage |
| 2399 | h2 | ■ Dados Técnicos para Análise | namespace and compile-time linkage |
| 2401 | h3 | Parâmetros de Simulação | namespace and compile-time linkage |
| 2411 | h3 | Limites de Detecção | namespace and compile-time linkage |
| 2418 | h3 | Dinâmica de Roda | namespace and compile-time linkage |
| 2427 | h2 | ■ Gráfico de Teste: Velocidade vs Tempo | namespace and compile-time linkage |
| 2429 | h3 | Teste de Freio Normal (sem ABS) | namespace and compile-time linkage |
| 2445 | h3 | Teste com ABS Ativo | namespace and compile-time linkage |
| 2463 | h2 | ■ Checklist de Validação | namespace and compile-time linkage |
| 2483 | h2 | ■ Troubleshooting | namespace and compile-time linkage |
| 2485 | h3 | "Cannot find module" ao compilar | namespace and compile-time linkage |
| 2487 | h1 | Solução: | namespace and compile-time linkage |
| 2493 | h2 | File Deep Dive: TUTORIALS.md | namespace and compile-time linkage |
| 2497 | h3 | Symbol Index | namespace and compile-time linkage |
| 2565 | h3 | Content Snapshot | namespace and compile-time linkage |
| 2567 | h1 | ■ Tutoriais e Exemplos - Sistema de Freio Autônomo com ABS | namespace and compile-time linkage |
| 2569 | h2 | ■ Sumário de Tutoriais | namespace and compile-time linkage |
| 2578 | h2 | ■ Iniciante: Primeiro Contato | namespace and compile-time linkage |
| 2580 | h3 | Objetivo | namespace and compile-time linkage |
| 2583 | h3 | Tutorial Passo-a-Passo | namespace and compile-time linkage |
| 2585 | h4 | Etapa 1: Iniciar a Simulação (1 minuto) | namespace and compile-time linkage |
| 2603 | h4 | Etapa 2: Aceleração Básica (2 minutos) | namespace and compile-time linkage |
| 2627 | h4 | Etapa 3: Freio Sem ABS (2 minutos) | namespace and compile-time linkage |
| 2653 | h4 | Etapa 4: Ativação do ABS (3 minutos) | namespace and compile-time linkage |
| 2684 | h4 | Etapa 5: Reset e Pausa (1 minuto) | namespace and compile-time linkage |
| 2705 | h3 | ■ Checklist Iniciante | namespace and compile-time linkage |
| 2717 | h2 | ■ Intermediário: Análise de Cenários | namespace and compile-time linkage |
| 2719 | h3 | Objetivo | namespace and compile-time linkage |
| 2724 | h3 | Teste 1: Modo Automático - Emergency Brake | namespace and compile-time linkage |
| 2726 | h4 | Preparação | namespace and compile-time linkage |
| 2730 | h4 | Timeline Esperada | namespace and compile-time linkage |
| 2742 | h4 | Coleta de Dados | namespace and compile-time linkage |
| 2758 | h4 | Análise de Resultados | namespace and compile-time linkage |
| 2776 | h3 | Teste 2: Modo Manual - Frenagem Progressiva | namespace and compile-time linkage |
| 2778 | h4 | Preparação | namespace and compile-time linkage |
| 2783 | h4 | Frenagem em Estágios | namespace and compile-time linkage |
| 2790 | h2 | File Deep Dive: allowlist.example.json | namespace and compile-time linkage |
| 2794 | h3 | Symbol Index | namespace and compile-time linkage |
| 2797 | h3 | Content Snapshot | namespace and compile-time linkage |
| 2817 | h2 | File Deep Dive: benches/solver_bench.rs | namespace and compile-time linkage |
| 2821 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 2829 | h3 | Symbol Index | namespace and compile-time linkage |
| 2834 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 2870 | h2 | File Deep Dive: docs/AI_AGENT_RUNBOOK.md | namespace and compile-time linkage |
| 2874 | h3 | Symbol Index | namespace and compile-time linkage |
| 2883 | h3 | Content Snapshot | namespace and compile-time linkage |
| 2885 | h1 | AI Agent Refactor Runbook | namespace and compile-time linkage |
| 2887 | h2 | Branch strategy | namespace and compile-time linkage |
| 2891 | h2 | Agent cycle | namespace and compile-time linkage |
| 2897 | h2 | Prompt template | namespace and compile-time linkage |

| 2905 | h2 | PR acceptance checklist | namespace and compile-time linkage |
| 2916 | h2 | File Deep Dive: docs/CAT_COMM_TEMPLATE_PROTOCOL.md | namespace and compile-time linkage |
| 2920 | h3 | Symbol Index | namespace and compile-time linkage |
| 2937 | h3 | Content Snapshot | namespace and compile-time linkage |
| 2939 | h1 | Cat Comm Template Bridge Protocol (Upload-Ready) | namespace and compile-time linkage |
| 2945 | h2 | 1. Startup modes | namespace and compile-time linkage |
| 2953 | h2 | 2. Transport contract | namespace and compile-time linkage |
| 2961 | h2 | 3. Request schema | namespace and compile-time linkage |
| 2965 | h3 | init | namespace and compile-time linkage |
| 2978 | h3 | ping | namespace and compile-time linkage |
| 2984 | h3 | read | namespace and compile-time linkage |
| 2993 | h3 | try_read | namespace and compile-time linkage |
| 2999 | h3 | send | namespace and compile-time linkage |
| 3031 | h3 | close | namespace and compile-time linkage |
| 3037 | h2 | 4. Response schema | namespace and compile-time linkage |
| 3108 | h2 | 5. Upload layout options | namespace and compile-time linkage |
| 3120 | h2 | 6. Safety expectations | namespace and compile-time linkage |
| 3130 | h2 | File Deep Dive: docs/DECISIONS.md | namespace and compile-time linkage |
| 3134 | h3 | Symbol Index | namespace and compile-time linkage |
| 3143 | h3 | Content Snapshot | namespace and compile-time linkage |
| 3145 | h1 | Architecture Decisions | namespace and compile-time linkage |
| 3147 | h2 | ADR-001: Incremental refactor with API preservation | namespace and compile-time linkage |
| 3151 | h2 | ADR-002: Feature flags for high-risk changes | namespace and compile-time linkage |
| 3155 | h2 | ADR-003: Test matrix-first workflow | namespace and compile-time linkage |
| 3159 | h2 | ADR-004: Structured telemetry baseline | namespace and compile-time linkage |
| 3164 | h2 | File Deep Dive: docs/ECM-Data.md | namespace and compile-time linkage |
| 3168 | h3 | Symbol Index | namespace and compile-time linkage |
| 3188 | h3 | Content Snapshot | namespace and compile-time linkage |
| 3190 | h1 | ECM-Data Production Runbook (Windows-first) | namespace and compile-time linkage |
| 3192 | h2 | 0. Executive summary | namespace and compile-time linkage |
| 3196 | h2 | 1. Goals and non-goals | namespace and compile-time linkage |
| 3210 | h2 | 2. Hardware adapter model | namespace and compile-time linkage |
| 3236 | h2 | 3. Safety controls | namespace and compile-time linkage |
| 3251 | h2 | 4. Allowlist schema and enforcement | namespace and compile-time linkage |
| 3270 | h2 | 5. Rate limiter sizing | namespace and compile-time linkage |
| 3288 | h2 | 6. Log and storage sizing | namespace and compile-time linkage |
| 3305 | h2 | 7. Metrics model | namespace and compile-time linkage |
| 3324 | h2 | 8. Failure modes and mitigations | namespace and compile-time linkage |
| 3339 | h2 | 9. Windows Cat Comm runbook | namespace and compile-time linkage |
| 3365 | h2 | 10. CI and test matrix | namespace and compile-time linkage |
| 3386 | h2 | 11. Security checklist | namespace and compile-time linkage |
| 3395 | h2 | 12. Upgrade roadmap | namespace and compile-time linkage |
| 3403 | h2 | 13. One-shot 12-phase execution | namespace and compile-time linkage |
| 3413 | h2 | File Deep Dive: docs/FULL_DESIGN_CODE_BOOK.md | namespace and compile-time linkage |
| 3417 | h3 | Symbol Index | namespace and compile-time linkage |
| 3607 | h3 | Content Snapshot | namespace and compile-time linkage |
| 3609 | h1 | AutoBreaking Full Design-Book and Code-Book | namespace and compile-time linkage |
| 3617 | h2 | Scope | namespace and compile-time linkage |
| 3624 | h2 | Build and Feature Context | namespace and compile-time linkage |
| 3632 | h2 | System Call-Flow (High Level) | namespace and compile-time linkage |
| 3640 | h2 | File Inventory | namespace and compile-time linkage |
| 3705 | h2 | File: src/adas.rs | namespace and compile-time linkage |
| 3710 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 3725 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 3832 | h2 | File Deep Dive: docs/OSS_INTEGRATION_MATRIX.md | namespace and compile-time linkage |
| 3836 | h3 | Symbol Index | namespace and compile-time linkage |
| 3857 | h3 | Content Snapshot | namespace and compile-time linkage |
| 3859 | h1 | OSS Integration Matrix for AutoBreaking | namespace and compile-time linkage |
| 3861 | h2 | Objective | namespace and compile-time linkage |
| 3872 | h2 | Selection Rules (hard gates) | namespace and compile-time linkage |
| 3880 | h2 | Candidate Matrix | namespace and compile-time linkage |
| 3892 | h2 | Module-to-OSS Mapping | namespace and compile-time linkage |
| 3894 | h2 | src/io/live_runner.rs | namespace and compile-time linkage |
| 3913 | h2 | src/io/vendor_cat_comm.rs | namespace and compile-time linkage |
| 3928 | h2 | src/io/production_program.rs | namespace and compile-time linkage |
| 3941 | h2 | src/io/hw.rs | namespace and compile-time linkage |
| 3958 | h2 | License and Compliance Notes | namespace and compile-time linkage |
| 3971 | h2 | Priority Adoption Plan | namespace and compile-time linkage |
| 3973 | h2 | Wave 1 (immediate, low risk) | namespace and compile-time linkage |
| 3979 | h2 | Wave 2 (short-term) | namespace and compile-time linkage |
| 3985 | h2 | Wave 3 (optional) | namespace and compile-time linkage |

```
| 3990 | h2 | Concrete Backlog Items | namespace and compile-time linkage |
| 4003 | h2 | Verification Gates for First Integration Patch Set | namespace and compile-time linkage |
| 4010 | h2 | External References (evaluated) | namespace and compile-time linkage |
| 4021 | h2 | File Deep Dive: fuzz/Cargo.toml | namespace and compile-time linkage |
| 4025 | h3 | Symbol Index | namespace and compile-time linkage |
| 4028 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 4051 | h2 | File Deep Dive: fuzz/fuzz_targets/j1939_frame_from_raw.rs | namespace and compile-time linkage |
| 4055 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 4064 | h3 | Symbol Index | namespace and compile-time linkage |
| 4067 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 4087 | h2 | File Deep Dive: reports/leak_predictions_20260812_095534.json | namespace and compile-time linkage |
| 4091 | h3 | Symbol Index | namespace and compile-time linkage |
| 4094 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4319 | h2 | File Deep Dive: reports/leak_report_20260812_095535.json | namespace and compile-time linkage |
| 4323 | h3 | Symbol Index | namespace and compile-time linkage |
| 4326 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4371 | h2 | File Deep Dive: reports/production_phase_report_1786621709828.json | namespace and compile-time linkage
| 4375 | h3 | Symbol Index | namespace and compile-time linkage |
| 4378 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4466 | h2 | File Deep Dive: reports/production_phase_report_1786621709828.md | namespace and compile-time linkage |
| 4470 | h3 | Symbol Index | namespace and compile-time linkage |
| 4476 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4478 | h1 | Production Program Report | namespace and compile-time linkage |
| 4485 | h2 | Phase Results | namespace and compile-time linkage |
| 4501 | h2 | File Deep Dive: reports/production_phase_report_1786621988419.json | namespace and compile-time linkage
| 4505 | h3 | Symbol Index | namespace and compile-time linkage |
| 4508 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4599 | h2 | File Deep Dive: reports/production_phase_report_1786621988419.md | namespace and compile-time linkage |
| 4603 | h3 | Symbol Index | namespace and compile-time linkage |
| 4609 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4611 | h1 | Production Program Report | namespace and compile-time linkage |
| 4618 | h2 | Phase Results | namespace and compile-time linkage |
| 4634 | h2 | File Deep Dive: reports/production_phase_report_1786622089893.json | namespace and compile-time linkage
| 4638 | h3 | Symbol Index | namespace and compile-time linkage |
| 4641 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4732 | h2 | File Deep Dive: reports/production_phase_report_1786622089893.md | namespace and compile-time linkage |
| 4736 | h3 | Symbol Index | namespace and compile-time linkage |
| 4742 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4744 | h1 | Production Program Report | namespace and compile-time linkage |
| 4751 | h2 | Phase Results | namespace and compile-time linkage |
| 4767 | h2 | File Deep Dive: reports/production_phase_report_1786622112533.json | namespace and compile-time linkage
| 4771 | h3 | Symbol Index | namespace and compile-time linkage |
| 4774 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4871 | h2 | File Deep Dive: reports/production_phase_report_1786622112533.md | namespace and compile-time linkage |
| 4875 | h3 | Symbol Index | namespace and compile-time linkage |
| 4881 | h3 | Content Snapshot | namespace and compile-time linkage |
| 4883 | h1 | Production Program Report | namespace and compile-time linkage |
| 4890 | h2 | Phase Results | namespace and compile-time linkage |
| 4906 | h2 | File Deep Dive: reports/production_phase_report_1786622154581.json | namespace and compile-time linkage
| 4910 | h3 | Symbol Index | namespace and compile-time linkage |
| 4913 | h3 | Content Snapshot | namespace and compile-time linkage |
| 5010 | h2 | File Deep Dive: reports/production_phase_report_1786622154581.md | namespace and compile-time linkage |
| 5014 | h3 | Symbol Index | namespace and compile-time linkage |
| 5020 | h3 | Content Snapshot | namespace and compile-time linkage |
| 5022 | h1 | Production Program Report | namespace and compile-time linkage |
| 5029 | h2 | Phase Results | namespace and compile-time linkage |
| 5045 | h2 | File Deep Dive: reports/production_phase_report_1786622968863.json | namespace and compile-time linkage
| 5049 | h3 | Symbol Index | namespace and compile-time linkage |
| 5052 | h3 | Content Snapshot | namespace and compile-time linkage |
| 5168 | h2 | File Deep Dive: reports/production_phase_report_1786622968863.md | namespace and compile-time linkage |
| 5172 | h3 | Symbol Index | namespace and compile-time linkage |
| 5179 | h3 | Content Snapshot | namespace and compile-time linkage |
| 5181 | h1 | Production Program Report | namespace and compile-time linkage |
| 5188 | h2 | UDS Conformance Summary | namespace and compile-time linkage |
| 5196 | h2 | Phase Results | namespace and compile-time linkage |
| 5212 | h2 | File Deep Dive: reports/production_phase_report_1786623531098.json | namespace and compile-time linkage
| 5216 | h3 | Symbol Index | namespace and compile-time linkage |
| 5219 | h3 | Content Snapshot | namespace and compile-time linkage |
| 5363 | h2 | File Deep Dive: reports/production_phase_report_1786623531098.md | namespace and compile-time linkage |
| 5367 | h3 | Symbol Index | namespace and compile-time linkage |
| 5374 | h3 | Content Snapshot | namespace and compile-time linkage |
```

| 5376 | h1 | Production Program Report | namespace and compile-time linkage |
| 5383 | h2 | UDS Conformance Summary | namespace and compile-time linkage |
| 5391 | h2 | Phase Results | namespace and compile-time linkage |
| 5407 | h2 | File Deep Dive: scripts/build_cat_comm_template_bridge.ps1 | namespace and compile-time linkage |
| 5411 | h3 | Symbol Index | namespace and compile-time linkage |
| 5414 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5448 | h2 | File Deep Dive: scripts/replay_from_artifact.sh | namespace and compile-time linkage |
| 5452 | h3 | Symbol Index | namespace and compile-time linkage |
| 5455 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5470 | h2 | File Deep Dive: scripts/run_fuzz_smoke.sh | namespace and compile-time linkage |
| 5474 | h3 | Symbol Index | namespace and compile-time linkage |
| 5477 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5490 | h2 | File Deep Dive: scripts/run_integration.sh | namespace and compile-time linkage |
| 5494 | h3 | Symbol Index | namespace and compile-time linkage |
| 5497 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5505 | h2 | File Deep Dive: scripts/run_monte_carlo.sh | namespace and compile-time linkage |
| 5509 | h3 | Symbol Index | namespace and compile-time linkage |
| 5512 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5533 | h2 | File Deep Dive: scripts/run_windows_template_e2e.ps1 | namespace and compile-time linkage |
| 5537 | h3 | Symbol Index | namespace and compile-time linkage |
| 5540 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5586 | h2 | File Deep Dive: scripts/validate_can_utils_interop.sh | namespace and compile-time linkage |
| 5590 | h3 | Symbol Index | namespace and compile-time linkage |
| 5593 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5660 | h2 | File Deep Dive: src/adas.rs | namespace and compile-time linkage |
| 5664 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 5674 | h3 | Symbol Index | namespace and compile-time linkage |
| 5688 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 5944 | h2 | File Deep Dive: src/autonomous.rs | namespace and compile-time linkage |
| 5948 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 5973 | h3 | Symbol Index | namespace and compile-time linkage |
| 6014 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 6665 | h2 | File Deep Dive: src/bin/cat_comm_bridge.rs | namespace and compile-time linkage |
| 6669 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 6687 | h3 | Symbol Index | namespace and compile-time linkage |
| 6706 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 7077 | h2 | File Deep Dive: src/bin/simulator_cli.rs | namespace and compile-time linkage |
| 7081 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 7099 | h3 | Symbol Index | namespace and compile-time linkage |
| 7107 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 7253 | h2 | File Deep Dive: src/boot_sequence.rs | namespace and compile-time linkage |
| 7257 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 7281 | h3 | Symbol Index | namespace and compile-time linkage |
| 7314 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 7835 | h2 | File Deep Dive: src/camera.rs | namespace and compile-time linkage |
| 7839 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 7854 | h3 | Symbol Index | namespace and compile-time linkage |
| 7896 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 8502 | h2 | File Deep Dive: src/can_bus.rs | namespace and compile-time linkage |
| 8506 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 8515 | h3 | Symbol Index | namespace and compile-time linkage |
| 8527 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 8711 | h2 | File Deep Dive: src/can_gateway.rs | namespace and compile-time linkage |
| 8715 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 8740 | h3 | Symbol Index | namespace and compile-time linkage |
| 8770 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 9091 | h2 | File Deep Dive: src/can_network.rs | namespace and compile-time linkage |
| 9095 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 9124 | h3 | Symbol Index | namespace and compile-time linkage |
| 9201 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 10586 | h2 | File Deep Dive: src/chassis.rs | namespace and compile-time linkage |
| 10590 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 10597 | h3 | Symbol Index | namespace and compile-time linkage |
| 10609 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 10797 | h2 | File Deep Dive: src/ecm_heavy.rs | namespace and compile-time linkage |
| 10801 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 10814 | h3 | Symbol Index | namespace and compile-time linkage |
| 10841 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 11221 | h2 | File Deep Dive: src/ecm_mock_provider.rs | namespace and compile-time linkage |
| 11225 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 11231 | h3 | Symbol Index | namespace and compile-time linkage |

| 11239 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 11441 | h2 | File Deep Dive: src/ecu_abs.rs | namespace and compile-time linkage |
| 11445 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 11471 | h3 | Symbol Index | namespace and compile-time linkage |
| 11512 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 12054 | h2 | File Deep Dive: src/ecu_bcm.rs | namespace and compile-time linkage |
| 12058 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 12076 | h3 | Symbol Index | namespace and compile-time linkage |
| 12100 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 12439 | h2 | File Deep Dive: src/ecu_ecm.rs | namespace and compile-time linkage |
| 12443 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 12469 | h3 | Symbol Index | namespace and compile-time linkage |
| 12511 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 13375 | h2 | File Deep Dive: src/ecu_hcm.rs | namespace and compile-time linkage |
| 13379 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 13398 | h3 | Symbol Index | namespace and compile-time linkage |
| 13423 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 13826 | h2 | File Deep Dive: src/ecu_icm.rs | namespace and compile-time linkage |
| 13830 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 13848 | h3 | Symbol Index | namespace and compile-time linkage |
| 13880 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 14351 | h2 | File Deep Dive: src/ecu_tcm.rs | namespace and compile-time linkage |
| 14355 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 14384 | h3 | Symbol Index | namespace and compile-time linkage |
| 14431 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 15151 | h2 | File Deep Dive: src/ecu_vcm.rs | namespace and compile-time linkage |
| 15155 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 15172 | h3 | Symbol Index | namespace and compile-time linkage |
| 15211 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 15637 | h2 | File Deep Dive: src/electrical.rs | namespace and compile-time linkage |
| 15641 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 15659 | h3 | Symbol Index | namespace and compile-time linkage |
| 15698 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 16159 | h2 | File Deep Dive: src/engine.rs | namespace and compile-time linkage |
| 16163 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 16171 | h3 | Symbol Index | namespace and compile-time linkage |
| 16185 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 16315 | h2 | File Deep Dive: src/gps.rs | namespace and compile-time linkage |
| 16319 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 16337 | h3 | Symbol Index | namespace and compile-time linkage |
| 16361 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 16732 | h2 | File Deep Dive: src/implement.rs | namespace and compile-time linkage |
| 16736 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 16755 | h3 | Symbol Index | namespace and compile-time linkage |
| 16795 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 17304 | h2 | File Deep Dive: src/imu.rs | namespace and compile-time linkage |
| 17308 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 17320 | h3 | Symbol Index | namespace and compile-time linkage |
| 17344 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 17695 | h2 | File Deep Dive: src/io/allowlist.rs | namespace and compile-time linkage |
| 17699 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 17714 | h3 | Symbol Index | namespace and compile-time linkage |
| 17733 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 17967 | h2 | File Deep Dive: src/io/artifact.rs | namespace and compile-time linkage |
| 17971 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 17979 | h3 | Symbol Index | namespace and compile-time linkage |
| 17985 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 18042 | h2 | File Deep Dive: src/io/ecm_params.rs | namespace and compile-time linkage |
| 18046 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 18057 | h3 | Symbol Index | namespace and compile-time linkage |
| 18072 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 18233 | h2 | File Deep Dive: src/io/hw.rs | namespace and compile-time linkage |
| 18237 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 18262 | h3 | Symbol Index | namespace and compile-time linkage |
| 18295 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 18704 | h2 | File Deep Dive: src/io/live_runner.rs | namespace and compile-time linkage |
| 18708 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 18737 | h3 | Symbol Index | namespace and compile-time linkage |
| 18806 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 20486 | h2 | File Deep Dive: src/io/metrics.rs | namespace and compile-time linkage |
| 20490 | h3 | Function Call Hotspots | namespace and compile-time linkage |

```
| 20502 | h3 | Symbol Index | namespace and compile-time linkage |
| 20516 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 20568 | h2 | File Deep Dive: src/io/mock.rs | namespace and compile-time linkage |
| 20572 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 20601 | h3 | Symbol Index | namespace and compile-time linkage |
| 20618 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 20809 | h2 | File Deep Dive: src/io/mod.rs | namespace and compile-time linkage |
| 20813 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 20818 | h3 | Symbol Index | namespace and compile-time linkage |
| 20835 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 20854 | h2 | File Deep Dive: src/io/production_program.rs | namespace and compile-time linkage |
| 20858 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 20887 | h3 | Symbol Index | namespace and compile-time linkage |
| 20909 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 21710 | h2 | File Deep Dive: src/io/rate_limiter.rs | namespace and compile-time linkage |
| 21714 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 21724 | h3 | Symbol Index | namespace and compile-time linkage |
| 21740 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 21829 | h2 | File Deep Dive: src/io/replay.rs | namespace and compile-time linkage |
| 21833 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 21850 | h3 | Symbol Index | namespace and compile-time linkage |
| 21865 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 22052 | h2 | File Deep Dive: src/io/serial_adapter.rs | namespace and compile-time linkage |
| 22056 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 22069 | h3 | Symbol Index | namespace and compile-time linkage |
| 22083 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 22183 | h2 | File Deep Dive: src/io/socketcan_adapter.rs | namespace and compile-time linkage |
| 22187 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 22200 | h3 | Symbol Index | namespace and compile-time linkage |
| 22217 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 22366 | h2 | File Deep Dive: src/io/vendor_cat_comm.rs | namespace and compile-time linkage |
| 22370 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 22395 | h3 | Symbol Index | namespace and compile-time linkage |
| 22426 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 22848 | h2 | File Deep Dive: src/j1939.rs | namespace and compile-time linkage |
| 22852 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 22881 | h3 | Symbol Index | namespace and compile-time linkage |
| 23014 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 24674 | h2 | File Deep Dive: src/leak_physics.rs | namespace and compile-time linkage |
| 24678 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 24707 | h3 | Symbol Index | namespace and compile-time linkage |
| 24806 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 26961 | h2 | File Deep Dive: src/lib.rs | namespace and compile-time linkage |
| 26965 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 26994 | h3 | Symbol Index | namespace and compile-time linkage |
| 27075 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 28248 | h2 | File Deep Dive: src/lidar.rs | namespace and compile-time linkage |
| 28252 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 28269 | h3 | Symbol Index | namespace and compile-time linkage |
| 28291 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 28651 | h2 | File Deep Dive: src/main.rs | namespace and compile-time linkage |
| 28655 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 28684 | h3 | Symbol Index | namespace and compile-time linkage |
| 28802 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 38272 | h2 | File Deep Dive: src/network_mgmt.rs | namespace and compile-time linkage |
| 38276 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 38294 | h3 | Symbol Index | namespace and compile-time linkage |
| 38321 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 38665 | h2 | File Deep Dive: src/nvm.rs | namespace and compile-time linkage |
| 38669 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 38698 | h3 | Symbol Index | namespace and compile-time linkage |
| 38742 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 39131 | h2 | File Deep Dive: src/observability.rs | namespace and compile-time linkage |
| 39135 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 39143 | h3 | Symbol Index | namespace and compile-time linkage |
| 39154 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 39209 | h2 | File Deep Dive: src/radar.rs | namespace and compile-time linkage |
| 39213 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 39228 | h3 | Symbol Index | namespace and compile-time linkage |
| 39257 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 39613 | h2 | File Deep Dive: src/sensors.rs | namespace and compile-time linkage |
```

```
| 39617 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 39626 | h3 | Symbol Index | namespace and compile-time linkage |
| 39636 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 39808 | h2 | File Deep Dive: src/sim_core.rs | namespace and compile-time linkage |
| 39812 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 39824 | h3 | Symbol Index | namespace and compile-time linkage |
| 39850 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 39925 | h2 | File Deep Dive: src/transmission.rs | namespace and compile-time linkage |
| 39929 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 39939 | h3 | Symbol Index | namespace and compile-time linkage |
| 39958 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 40103 | h2 | File Deep Dive: src/uds.rs | namespace and compile-time linkage |
| 40107 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 40136 | h3 | Symbol Index | namespace and compile-time linkage |
| 40212 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 41178 | h2 | File Deep Dive: src/v2x_telematics.rs | namespace and compile-time linkage |
| 41182 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 41196 | h3 | Symbol Index | namespace and compile-time linkage |
| 41234 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 41872 | h2 | File Deep Dive: src/vehicle.rs | namespace and compile-time linkage |
| 41876 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 41888 | h3 | Symbol Index | namespace and compile-time linkage |
| 41907 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 42059 | h2 | File Deep Dive: src/widgets.rs | namespace and compile-time linkage |
| 42063 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 42073 | h3 | Symbol Index | namespace and compile-time linkage |
| 42085 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 42416 | h2 | File Deep Dive: tests/io_mock.rs | namespace and compile-time linkage |
| 42420 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 42444 | h3 | Symbol Index | namespace and compile-time linkage |
| 42457 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 42669 | h2 | File Deep Dive: tests/leak_system_integration.rs | namespace and compile-time linkage |
| 42673 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 42692 | h3 | Symbol Index | namespace and compile-time linkage |
| 42698 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 42786 | h2 | File Deep Dive: tests/property_invariants.rs | namespace and compile-time linkage |
| 42790 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 42800 | h3 | Symbol Index | namespace and compile-time linkage |
| 42806 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 42848 | h2 | File Deep Dive: tests/speed_regression.rs | namespace and compile-time linkage |
| 42852 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 42864 | h3 | Symbol Index | namespace and compile-time linkage |
| 42872 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 42942 | h2 | File Deep Dive: tests/system_failure_scenarios.rs | namespace and compile-time linkage |
| 42946 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 42962 | h3 | Symbol Index | namespace and compile-time linkage |
| 42972 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 43057 | h2 | File Deep Dive: tests/vendor_bridge_e2e.rs | namespace and compile-time linkage |
| 43061 | h3 | Function Call Hotspots | namespace and compile-time linkage |
| 43074 | h3 | Symbol Index | namespace and compile-time linkage |
| 43081 | h3 | Line by Line Change Impact | namespace and compile-time linkage |
| 43166 | h2 | Cross File Traceability | namespace and compile-time linkage |
| 43172 | h2 | Change Simulation Playbook | namespace and compile-time linkage |
| 43178 | h2 | Full Raw Context Appendix | namespace and compile-time linkage |

### Content Snapshot
```

# AutoBreaking Ultimate Full Book

> Complete design-book and code-book generated from repository state with maximal practical coverage.

## Meta

- Generated at: 2026-08-13 09:41:06
- Git commit: 237eebc

- Files covered: 107
- Rust files covered: 62

## Architecture Narrative

1. Core simulation and ECU domain logic live under src and feed deterministic/runtime scenarios.
2. Live hardware and flashing orchestration live in src/io with strict protocol checks and gating.
3. Binary entrypoints in src/bin drive CLI workflows and template bridge interactions.
4. tests contains regression, property, and integration suites including vendor bridge e2e.
5. scripts and .github files define operational and CI automation.

## Module Coupling Graph

```
graph LR
  adas --> sensors
  boot_sequence --> can_gateway
  boot_sequence --> j1939
  can_gateway --> j1939
  can_network --> j1939
  ecm_heavy --> j1939
  ecm_mock_provider --> io
  ecu_abs --> j1939
  ecu_bcm --> j1939
  ecu_ecm --> j1939
  ecu_hcm --> j1939
  ecu_icm --> j1939
  ecu_tcm --> j1939
  ecu_vcm --> j1939
  electrical --> IgnitionState
  implement --> j1939
  io_allowlist --> io
  io_live_runner --> io
  io_production_program --> HeavyMachinery
  sensors --> vehicle
  uds --> j1939
```

## Dependency Tree Snapshot

```
auto_breaking v0.3.0 (C:\Users\costabr\OneDrive - Caterpillar\Documents\Rust\AutoBreaking)
â”œâ”€â” chrono v0.4.45
â”,   â”œâ”€â” num-traits v0.2.19
â”,   â”,   [build-dependencies]
â”,   â”,   â””â”€â” autocfg v1.5.1
â”,   â””â”€â” windows-link v0.2.1
â”œâ”€â” crc32fast v1.5.0
â”,   â””â”€â” cfg-if v1.0.4
â”œâ”€â” csv v1.4.0
â”,   â”œâ”€â” csv-core v0.1.13
â”,   â”,   â””â”€â” memchr v2.8.3
â”,   â”œâ”€â” itoa v1.0.18
â”,   â”œâ”€â” ryu v1.0.23
â”,   â””â”€â” serde_core v1.0.229
â”œâ”€â” eframe v0.29.1
â”,   â”œâ”€â” ahash v0.8.12
â”,   â”,   â”œâ”€â” cfg-if v1.0.4
â”,   â”,   â”œâ”€â” once_cell v1.21.4
â”,   â”,   â””â”€â” zerocopy v0.8.56
â”,   â”,       â””â”€â” zerocopy-derive v0.8.56 (proc-macro)
â”,   â”,           â”œâ”€â” proc-macro2 v1.0.107
â”,   â”,           â”,   â””â”€â” unicode-ident v1.0.24
â”,   â”,           â”œâ”€â” quote v1.0.47
â”,   â”,           â”,   â””â”€â” proc-macro2 v1.0.107 (*)
```

```
â”‚   â”‚           â””â”€â”€ syn v2.0.119
â”‚   â”‚               â”œâ”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚               â”œâ”€â”€ quote v1.0.47 (*)
â”‚   â”‚               â””â”€â”€ unicode-ident v1.0.24
â”‚   â”‚   [build-dependencies]
â”‚   â”‚   â””â”€â”€ version_check v0.9.5
â”‚   â”œâ”€â”€ document-features v0.2.12 (proc-macro)
â”‚   â”‚   â””â”€â”€ litrs v1.0.0
â”‚   â”œâ”€â”€ egui v0.29.1
â”‚   â”‚   â”œâ”€â”€ accesskit v0.16.3
â”‚   â”‚   â”œâ”€â”€ ahash v0.8.12 (*)
â”‚   â”‚   â”œâ”€â”€ emath v0.29.1
â”‚   â”‚   â│   â””â”€â”€ bytemuck v1.25.2
â”‚   â”‚   â│       â””â”€â”€ bytemuck_derive v1.12.0 (proc-macro)
â”‚   â”‚   â│           â”œâ”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚   â│           â├â”€â”€ quote v1.0.47 (*)
â”‚   â”‚   â│           â””â”€â”€ syn v3.0.3
â”‚   â”‚   â│               â├â”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚   â│               â├â”€â”€ quote v1.0.47 (*)
â”‚   â”‚   â│               â””â”€â”€ unicode-ident v1.0.24
â”‚   â”‚   â├â”€â”€ epaint v0.29.1
â”‚   â│   â│   â├â”€â”€ ab_glyph v0.2.32
â”‚   â│   â│   â│   â├â”€â”€ ab_glyph_rasterizer v0.1.10
â”‚   â│   â│   â│   â│   â””â”€â”€ owned_ttf_parser v0.25.1
â”‚   â│   â│   â│   â│       â””â”€â”€ ttf-parser v0.25.1
â”‚   â│   â│   â├â”€â”€ ahash v0.8.12 (*)
â”‚   â│   â│   â├â”€â”€ bytemuck v1.25.2 (*)
â”‚   â│   â│   â├â”€â”€ ecolor v0.29.1
â”‚   â│   â│   â│   â├â”€â”€ bytemuck v1.25.2 (*)
â”‚   â│   â│   â│   â│   â””â”€â”€ emath v0.29.1 (*)
â”‚   â│   â│   â├â”€â”€ emath v0.29.1 (*)
â”‚   â│   â│   â├â”€â”€ epaint_default_fonts v0.29.1
â”‚   â│   â│   â├â”€â”€ log v0.4.33
â”‚   â│   â│   â├â”€â”€ nohash-hasher v0.2.0
â”‚   â│   â│   â””â”€â”€ parking_lot v0.12.5
â”‚   â│   â│       â├â”€â”€ lock_api v0.4.14
â”‚   â│   â│       â│   â””â”€â”€ scopeguard v1.2.0
â”‚   â│   â│       â””â”€â”€ parking_lot_core v0.9.12
â”‚   â│   â│           â├â”€â”€ cfg-if v1.0.4
â”‚   â│   â│           â├â”€â”€ smallvec v1.15.2
â”‚   â│   â│           â””â”€â”€ windows-link v0.2.1
â”‚   â│   â├â”€â”€ log v0.4.33
â”‚   â│   â””â”€â”€ nohash-hasher v0.2.0
â”‚   â├â”€â”€ egui-winit v0.29.1
â”‚   â│   â├â”€â”€ accesskit_winit v0.22.4
â”‚   â│   â│   â├â”€â”€ accesskit v0.16.3
â”‚   â│   â│   â├â”€â”€ accesskit_windows v0.23.2
â”‚   â│   â│   â│   â├â”€â”€ accesskit v0.16.3
â”‚   â│   â│   â│   â├â”€â”€ accesskit_consumer v0.24.3
â”‚   â│   â│   â│   â│   â├â”€â”€ accesskit v0.16.3
â”‚   â│   â│   â│   â│   â│   â””â”€â”€ immutable-chunkmap v2.1.3
â”‚   â│   â│   â│   â│   â│       â””â”€â”€ arrayvec v0.7.8
â”‚   â│   â│   â│   â│   â├â”€â”€ paste v1.0.15 (proc-macro)
â”‚   â│   â│   â│   â│   â├â”€â”€ static_assertions v1.1.0
â”‚   â│   â│   â│   â│   â├â”€â”€ windows v0.58.0
â”‚   â│   â│   â│   â│   â│   â├â”€â”€ windows-core v0.58.0
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ windows-implement v0.58.0 (proc-macro)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ quote v1.0.47 (*)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ syn v2.0.119 (*)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ windows-interface v0.58.0 (proc-macro)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ quote v1.0.47 (*)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ syn v2.0.119 (*)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ windows-result v0.2.0
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ windows-targets v0.52.6
â”‚   â│   â│   â│   â│   â│   â│   â│       â””â”€â”€ windows_x86_64_msvc v0.52.6
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ windows-strings v0.1.0
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ windows-result v0.2.0 (*)
â”‚   â│   â│   â│   â│   â│   â│   â├â”€â”€ windows-targets v0.52.6 (*)
â”‚   â│   â│   â│   â│   â│   â│   â””â”€â”€ windows-targets v0.52.6 (*)
```

```
â”‚   â”‚   â”‚   â”‚   â”‚   â””â”€â”€ windows-targets v0.52.6 (*)
â”‚   â”‚   â”‚   â”‚   â””â”€â”€ windows-core v0.58.0 (*)
â”‚   â”‚   â”‚   â”œâ”€â”€ raw-window-handle v0.6.2
â”‚   â”‚   â”‚   â””â”€â”€ winit v0.30.13
â”‚   â”‚   â”‚       â”œâ”€â”€ bitflags v2.13.1
â”‚   â”‚   â”‚       â”œâ”€â”€ cursor-icon v1.2.0
â”‚   â”‚   â”‚       â”œâ”€â”€ dpi v0.1.2
â”‚   â”‚   â”‚       â”œâ”€â”€ raw-window-handle v0.6.2
â”‚   â”‚   â”‚       â”œâ”€â”€ smol_str v0.2.2
â”‚   â”‚   â”‚       â”œâ”€â”€ tracing v0.1.44
â”‚   â”‚   â”‚       â”‚   â”œâ”€â”€ pin-project-lite v0.2.17
â”‚   â”‚   â”‚       â”‚   â””â”€â”€ tracing-core v0.1.36
â”‚   â”‚   â”‚       â”œâ”€â”€ unicode-segmentation v1.13.3
â”‚   â”‚   â”‚       â””â”€â”€ windows-sys v0.52.0
â”‚   â”‚   â”‚           â””â”€â”€ windows-targets v0.52.6 (*)
â”‚   â”‚   â”‚       [build-dependencies]
â”‚   â”‚   â”‚       â””â”€â”€ cfg_aliases v0.2.2
â”‚   â”‚   â”œâ”€â”€ ahash v0.8.12 (*)
â”‚   â”‚   â”œâ”€â”€ arboard v3.6.1
â”‚   â”‚   â”‚   â”œâ”€â”€ clipboard-win v5.4.1
â”‚   â”‚   â”‚   â”‚   â””â”€â”€ error-code v3.3.2
â”‚   â”‚   â”‚   â”œâ”€â”€ log v0.4.33
â”‚   â”‚   â”‚   â””â”€â”€ windows-sys v0.60.2
â”‚   â”‚   â”‚       â””â”€â”€ windows-targets v0.53.5
â”‚   â”‚   â”‚           â””â”€â”€ windows_x86_64_msvc v0.53.1
â”‚   â”‚   â”œâ”€â”€ egui v0.29.1 (*)
â”‚   â”‚   â”œâ”€â”€ log v0.4.33
â”‚   â”‚   â”œâ”€â”€ raw-window-handle v0.6.2
â”‚   â”‚   â”œâ”€â”€ web-time v1.1.0
â”‚   â”‚   â”œâ”€â”€ webbrowser v1.2.4
â”‚   â”‚   â”‚   â”œâ”€â”€ log v0.4.33
â”‚   â”‚   â”‚   â””â”€â”€ url v2.5.8
â”‚   â”‚   â”‚       â”œâ”€â”€ form_urlencoded v1.2.2
â”‚   â”‚   â”‚       â”‚   â””â”€â”€ percent-encoding v2.3.2
â”‚   â”‚   â”‚       â”œâ”€â”€ idna v1.1.0
â”‚   â”‚   â”‚       â”‚   â”œâ”€â”€ idna_adapter v1.2.2
â”‚   â”‚   â”‚       â”‚   â”‚   â”œâ”€â”€ icu_normalizer v2.2.0
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”œâ”€â”€ icu_collections v2.2.0
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”œâ”€â”€ displaydoc v0.2.7 (proc-macro)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚   â”œâ”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚   â”œâ”€â”€ quote v1.0.47 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚   â””â”€â”€ syn v3.0.3 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”œâ”€â”€ potential_utf v0.1.5
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚   â””â”€â”€ zerovec v0.11.6
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”œâ”€â”€ yoke v0.8.3
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”œâ”€â”€ stable_deref_trait v1.2.1
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”œâ”€â”€ yoke-derive v0.8.2 (proc-macro)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”‚   â”œâ”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”‚   â”œâ”€â”€ quote v1.0.47 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”‚   â”œâ”€â”€ syn v2.0.119 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”‚   â””â”€â”€ synstructure v0.13.2
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”‚       â”œâ”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”‚       â”œâ”€â”€ quote v1.0.47 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â”‚       â””â”€â”€ syn v2.0.119 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚   â””â”€â”€ zerofrom v0.1.8
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚       â””â”€â”€ zerofrom-derive v0.1.7 (proc-macro)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚           â”œâ”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚           â”œâ”€â”€ quote v1.0.47 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚           â”œâ”€â”€ syn v2.0.119 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”‚           â””â”€â”€ synstructure v0.13.2 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â”œâ”€â”€ zerofrom v0.1.8 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚       â””â”€â”€ zerovec-derive v0.11.3 (proc-macro)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚           â”œâ”€â”€ proc-macro2 v1.0.107 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚           â”œâ”€â”€ quote v1.0.47 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”‚           â””â”€â”€ syn v2.0.119 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”œâ”€â”€ utf8_iter v1.0.4
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”œâ”€â”€ yoke v0.8.3 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â”œâ”€â”€ zerofrom v0.1.8 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”‚   â””â”€â”€ zerovec v0.11.6 (*)
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”œâ”€â”€ icu_normalizer_data v2.2.0
â”‚   â”‚   â”‚       â”‚   â”‚   â”‚   â”œâ”€â”€ icu_provider v2.2.0
```

```
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”œâ”€â”€ displaydoc v0.2.7 (proc-macro) (*)
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”œâ”€â”€ icu_locale_core v2.2.0
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”,   â”œâ”€â”€ displaydoc v0.2.7 (proc-macro) (*)
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”,   â”œâ”€â”€ litemap v0.8.2
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”,   â”œâ”€â”€ tinystr v0.8.3
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”,   â”,   â”œâ”€â”€ displaydoc v0.2.7 (proc-macro) (*)
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”,   â”,   â””â”€â”€ zerovec v0.11.6 (*)
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”,   â”œâ”€â”€ writeable v0.6.3
â”,   â”,   â”,       â”,   â”,   â”,   â”,   â”,   â””â”€â”€ zerovec v0.11.6 (*)
... truncated 42959 lines for this snapshot in markdown volume ...
```

# File Deep Dive: docs/FULL_BOOK_V2_CHANGE_ENGINEERING.md

Role: human-facing architecture, protocols, runbooks, and design records

Line count: 294 | Non-empty: 282 | Symbol count: 10

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | AutoBreaking V2 Change Engineering Book | namespace and compile-time linkage |
| 5 | h2 | 1. Matriz de Risco por Funcao Critica | namespace and compile-time linkage |
| 128 | h2 | 2. Se Mudar X Entao Quebra Y | namespace and compile-time linkage |
| 251 | h2 | 3. Roadmap de Refatoracao Segura por Ondas | namespace and compile-time linkage |
| 253 | h3 | Wave 1 - Blindagem de Contratos de Protocolo | namespace and compile-time linkage |
| 261 | h3 | Wave 2 - Refino de Abstracao de Hardware | namespace and compile-time linkage |
| 269 | h3 | Wave 3 - Estabilizacao de Schema de Relatorio | namespace and compile-time linkage |
| 277 | h3 | Wave 4 - Performance e Observabilidade | namespace and compile-time linkage |
| 284 | h3 | Wave 5 - Harden de CI e Evidencia | namespace and compile-time linkage |
| 292 | h2 | Cobertura e Limites | namespace and compile-time linkage |
```

## Content Snapshot

```
# AutoBreaking V2 Change Engineering Book

Gerado em: 2026-08-13 09:44:05

## 1. Matriz de Risco por Funcao Critica

| Risco | Funcao | Local | Linha | Justificativa | Testes Minimos |
|---|---|---|---:|---|---|
| critical | handle_uds_serial_request | src/bin/cat_comm_bridge.rs | 101 | impacts diagnostic protocol contract and E
| critical | missing_for_live_flash | src/io/hw.rs | 49 | impacts firmware write path, transfer integrity, and product
| critical | declared_capabilities | src/io/hw.rs | 231 | impacts hardware transport abstraction and live connectivity
| critical | open_real_adapter | src/io/hw.rs | 298 | impacts hardware transport abstraction and live connectivity | c
| critical | probe_live_adapter | src/io/hw.rs | 324 | impacts hardware transport abstraction and live connectivity |
| critical | append_flash_audit | src/io/live_runner.rs | 484 | impacts firmware write path, transfer integrity, and p
| critical | run_flash_preflight | src/io/live_runner.rs | 749 | impacts firmware write path, transfer integrity, and
| critical | j1939_uds_req_id | src/io/live_runner.rs | 816 | impacts diagnostic protocol contract and ECU session sta
| critical | is_uds_response_id | src/io/live_runner.rs | 820 | impacts diagnostic protocol contract and ECU session s
| critical | recv_can_uds_response | src/io/live_runner.rs | 1046 | impacts diagnostic protocol contract and ECU sessi
| critical | send_uds_and_wait | src/io/live_runner.rs | 1164 | impacts diagnostic protocol contract and ECU session s
| critical | live_flash_ecm_firmware | src/io/live_runner.rs | 1286 | impacts firmware write path, transfer integrity,
| critical | uds_response_id_matches_expected_target | src/io/live_runner.rs | 1631 | impacts diagnostic protocol cont
| critical | uds_response_id_rejects_wrong_pf_ps_or_sa | src/io/live_runner.rs | 1639 | impacts diagnostic protocol co
| critical | run_all_phases | src/io/production_program.rs | 77 | impacts release gate and conformance reporting | car
| critical | run_sim_identity_check | src/io/production_program.rs | 601 | impacts release gate and conformance report
| critical | build_conformance_summary | src/io/production_program.rs | 613 | impacts release gate and conformance rep
| critical | run_uds_runtime_self_test | src/io/production_program.rs | 744 | impacts diagnostic protocol contract and
| critical | supports | src/io/vendor_cat_comm.rs | 30 | impacts hardware transport abstraction and live connectivity
| critical | open | src/io/vendor_cat_comm.rs | 88 | impacts hardware transport abstraction and live connectivity | ca
| critical | ensure_connected | src/io/vendor_cat_comm.rs | 154 | impacts hardware transport abstraction and live conn
| critical | send_bridge_request | src/io/vendor_cat_comm.rs | 164 | impacts hardware transport abstraction and live c
| critical | read_bridge_response | src/io/vendor_cat_comm.rs | 182 | impacts hardware transport abstraction and live
| critical | append_diag_record | src/io/vendor_cat_comm.rs | 201 | impacts hardware transport abstraction and live co
| critical | init | src/io/vendor_cat_comm.rs | 233 | impacts hardware transport abstraction and live connectivity | c
| critical | read_frame | src/io/vendor_cat_comm.rs | 246 | impacts hardware transport abstraction and live connectivi
| critical | try_read_frame | src/io/vendor_cat_comm.rs | 261 | impacts hardware transport abstraction and live connec
| critical | send_frame | src/io/vendor_cat_comm.rs | 273 | impacts hardware transport abstraction and live connectivi
```

```
| critical | close | src/io/vendor_cat_comm.rs | 286 | impacts hardware transport abstraction and live connectivity |
| critical | adapter_info | src/io/vendor_cat_comm.rs | 305 | impacts hardware transport abstraction and live connecti
| critical | resolve_bridge_exe | src/io/vendor_cat_comm.rs | 318 | impacts hardware transport abstraction and live co
| critical | map_bridge_error | src/io/vendor_cat_comm.rs | 347 | impacts hardware transport abstraction and live conn
| critical | bridge_info_from_response | src/io/vendor_cat_comm.rs | 368 | impacts hardware transport abstraction and
| critical | ensure_bridge_compatibility | src/io/vendor_cat_comm.rs | 375 | impacts hardware transport abstraction an
| critical | now_ms | src/io/vendor_cat_comm.rs | 413 | impacts hardware transport abstraction and live connectivity |
| critical | engineering_oil_catalog | src/leak_physics.rs | 496 | impacts runtime behavior and downstream callers | c
| critical | uds_process_sa | src/main.rs | 6236 | impacts diagnostic protocol contract and ECU session state | cargo
| critical | uds_send_and_log | src/main.rs | 6246 | impacts diagnostic protocol contract and ECU session state | carg
| critical | run_uds_flash_pipeline | src/main.rs | 6278 | impacts firmware write path, transfer integrity, and produc
| critical | tab_uds | src/main.rs | 9040 | impacts diagnostic protocol contract and ECU session state | cargo check ;
| critical | fmt | src/uds.rs | 42 | impacts runtime behavior and downstream callers | cargo check ; cargo check --fea
| critical | new | src/uds.rs | 237 | impacts runtime behavior and downstream callers | cargo check ; cargo check --fe
| critical | tick | src/uds.rs | 276 | impacts runtime behavior and downstream callers | cargo check ; cargo check --f
| critical | process | src/uds.rs | 293 | impacts runtime behavior and downstream callers | cargo check ; cargo check
| critical | svc_session_control | src/uds.rs | 320 | impacts runtime behavior and downstream callers | cargo check ;
| critical | svc_ecu_reset | src/uds.rs | 353 | impacts runtime behavior and downstream callers | cargo check ; cargo
| critical | svc_clear_dtc | src/uds.rs | 395 | impacts runtime behavior and downstream callers | cargo check ; cargo
| critical | svc_read_dtc | src/uds.rs | 428 | impacts runtime behavior and downstream callers | cargo check ; cargo c
| critical | svc_read_data_by_id | src/uds.rs | 504 | impacts runtime behavior and downstream callers | cargo check ;
| critical | svc_security_access | src/uds.rs | 554 | impacts diagnostic protocol contract and ECU session state | car
| critical | svc_write_data_by_id | src/uds.rs | 640 | impacts runtime behavior and downstream callers | cargo check ;
| critical | svc_routine_control | src/uds.rs | 684 | impacts runtime behavior and downstream callers | cargo check ;
| critical | svc_request_download | src/uds.rs | 733 | impacts firmware write path, transfer integrity, and production
| critical | svc_transfer_data | src/uds.rs | 767 | impacts firmware write path, transfer integrity, and production ga
| critical | svc_request_transfer_exit | src/uds.rs | 809 | impacts firmware write path, transfer integrity, and produ
| critical | svc_tester_present | src/uds.rs | 830 | impacts runtime behavior and downstream callers | cargo check ; c
| critical | record_dtc_set | src/uds.rs | 850 | impacts runtime behavior and downstream callers | cargo check ; cargo
| critical | nrc | src/uds.rs | 879 | impacts runtime behavior and downstream callers | cargo check ; cargo check --fe
| critical | log | src/uds.rs | 883 | impacts runtime behavior and downstream callers | cargo check ; cargo check --fe
| critical | service_name | src/uds.rs | 905 | impacts runtime behavior and downstream callers | cargo check ; cargo c
| critical | dtc_to_3bytes | src/uds.rs | 926 | impacts runtime behavior and downstream callers | cargo check ; cargo
| critical | session_default_relocks_security | src/uds.rs | 939 | impacts diagnostic protocol contract and ECU sessio
| critical | transfer_exit_rejects_incomplete_download | src/uds.rs | 949 | impacts firmware write path, transfer inte
| critical | bin_path | tests/vendor_bridge_e2e.rs | 9 | impacts hardware transport abstraction and live connectivity
| critical | unique_temp_dir | tests/vendor_bridge_e2e.rs | 22 | impacts hardware transport abstraction and live conne
| critical | cat_comm_bridge_probe_and_live_phases_flash_e2e | tests/vendor_bridge_e2e.rs | 33 | impacts firmware writ
| high | from_path | src/io/allowlist.rs | 33 | impacts runtime behavior and downstream callers | cargo check ; cargo
| high | is_allowed | src/io/allowlist.rs | 42 | impacts runtime behavior and downstream callers | cargo check ; cargo
| high | per_rule_rate_limit | src/io/allowlist.rs | 46 | impacts runtime behavior and downstream callers | cargo chec
| high | validate | src/io/allowlist.rs | 53 | impacts runtime behavior and downstream callers | cargo check ; cargo t
| high | rule_matches_frame | src/io/allowlist.rs | 95 | impacts runtime behavior and downstream callers | cargo check
| high | parse_u32_hex_or_dec | src/io/allowlist.rs | 137 | impacts runtime behavior and downstream callers | cargo ch
| high | parse_mask | src/io/allowlist.rs | 146 | impacts runtime behavior and downstream callers | cargo check ; carg
| high | match_hex_pattern | src/io/allowlist.rs | 150 | impacts runtime behavior and downstream callers | cargo check
| high | can_rule_matches_mask | src/io/allowlist.rs | 182 | impacts runtime behavior and downstream callers | cargo c
| high | serial_rule_matches_pattern | src/io/allowlist.rs | 209 | impacts runtime behavior and downstream callers | c
| high | write_intent_enabled | src/io/hw.rs | 223 | impacts hardware transport abstraction and live connectivity | ca
| high | connect_ecm | src/io/live_runner.rs | 174 | impacts runtime behavior and downstream callers | cargo check ; c
| high | live_preflight_ecm | src/io/live_runner.rs | 793 | impacts runtime behavior and downstream callers | cargo ch
| high | check_live_channel_policy | src/io/live_runner.rs | 1216 | impacts runtime behavior and downstream callers |
| high | on_rate_limited | src/io/metrics.rs | 37 | impacts runtime behavior and downstream callers | cargo check ; ca
| high | inject_disconnect | src/io/mock.rs | 32 | impacts runtime behavior and downstream callers | cargo check ; car
| high | clear_disconnect | src/io/mock.rs | 36 | impacts runtime behavior and downstream callers | cargo check ; carg
| high | read_frame | src/io/mock.rs | 61 | impacts runtime behavior and downstream callers | cargo check ; cargo test
| high | try_read_frame | src/io/mock.rs | 89 | impacts runtime behavior and downstream callers | cargo check ; cargo
| high | send_frame | src/io/mock.rs | 114 | impacts runtime behavior and downstream callers | cargo check ; cargo tes
| high | default | src/io/production_program.rs | 69 | impacts release gate and conformance reporting | cargo check ;
| high | run_sim_baseline_check | src/io/production_program.rs | 568 | impacts release gate and conformance reporting
| high | run_sim_connectivity_check | src/io/production_program.rs | 587 | impacts release gate and conformance report
| high | run_sim_preflight_check | src/io/production_program.rs | 731 | impacts release gate and conformance reporting
| high | run_sim_stress_check | src/io/production_program.rs | 763 | impacts release gate and conformance reporting |
| high | hex_sha | src/io/production_program.rs | 784 | impacts release gate and conformance reporting | cargo check ;
| high | now_ms | src/io/production_program.rs | 792 | impacts release gate and conformance reporting | cargo check ;
| high | new | src/io/rate_limiter.rs | 13 | impacts runtime behavior and downstream callers | cargo check ; cargo tes
| high | try_take | src/io/rate_limiter.rs | 23 | impacts runtime behavior and downstream callers | cargo check ; carg
| high | refill | src/io/rate_limiter.rs | 33 | impacts runtime behavior and downstream callers | cargo check ; cargo
| high | new | src/io/rate_limiter.rs | 49 | impacts runtime behavior and downstream callers | cargo check ; cargo tes
| high | check_can | src/io/rate_limiter.rs | 57 | impacts runtime behavior and downstream callers | cargo check ; car
| high | check_serial | src/io/rate_limiter.rs | 70 | impacts runtime behavior and downstream callers | cargo check ;
```

| high | limiter_blocks_when_empty | src/io/rate_limiter.rs | 80 | impacts runtime behavior and downstream callers | c
| high | aggregate_group_report | src/leak_physics.rs | 73 | impacts evidence schema consumed by automation and approv
| high | export_calibration_report_json | src/leak_physics.rs | 1654 | impacts evidence schema consumed by automation
| high | export_calibration_report_csv | src/leak_physics.rs | 1670 | impacts evidence schema consumed by automation a
| high | report_captures_pressure_band_and_safe_window | src/leak_physics.rs | 2030 | impacts evidence schema consumed
| high | calibration_mode_ingests_csv_and_exports_reports | src/leak_physics.rs | 2107 | impacts evidence schema consu
| high | export_leak_report_json | src/lib.rs | 1053 | impacts evidence schema consumed by automation and approvals |
| high | export_leak_report_csv | src/lib.rs | 1060 | impacts evidence schema consumed by automation and approvals | c
| high | export_leak_calibration_report_json | src/lib.rs | 1118 | impacts evidence schema consumed by automation and
| high | export_leak_calibration_report_csv | src/lib.rs | 1126 | impacts evidence schema consumed by automation and a
| high | pick_save_path | src/main.rs | 542 | impacts runtime behavior and downstream callers | cargo check ; cargo te
| high | leak_ascii_cad | src/main.rs | 6360 | impacts runtime behavior and downstream callers | cargo check ; cargo t
| high | fmt | src/v2x_telematics.rs | 322 | impacts runtime behavior and downstream callers | cargo check ; cargo tes
| high | write_test_allowlist | tests/io_mock.rs | 10 | impacts write authorization and operational safety | cargo che
| high | mock_adapter_allows_write_when_policy_satisfied | tests/io_mock.rs | 97 | impacts hardware transport abstract
| high | mock_adapter_dry_run_logs_without_physical_tx | tests/io_mock.rs | 125 | impacts hardware transport abstracti
| high | mock_adapter_rate_limit_blocks_burst | tests/io_mock.rs | 157 | impacts hardware transport abstraction and li
| high | allowlist_parse_mask_accepts_hex | tests/io_mock.rs | 189 | impacts write authorization and operational safet
| high | e2e_hydraulic_plus_ac_stress_generates_reports | tests/leak_system_integration.rs | 4 | impacts evidence sche

## 2. Se Mudar X Entao Quebra Y

| X alterado | Onde | Risco | Y afetado | Linhas de referencia | Efeito esperado |
|---|---|---|---|---|---|
| handle_uds_serial_request | src/bin/cat_comm_bridge.rs | critica | chamadas diretas nao detectadas por varredura est
| missing_for_live_flash | src/io/hw.rs | critica | src/io/production_program.rs ; src/io/production_program.rs ; src/
| declared_capabilities | src/io/hw.rs | critica | src/io/production_program.rs | src/io/hw.rs:231 ; src/io/production
| open_real_adapter | src/io/hw.rs | critica | src/io/live_runner.rs ; src/io/live_runner.rs ; src/io/live_runner.rs ;
| probe_live_adapter | src/io/hw.rs | critica | src/bin/simulator_cli.rs ; tests/vendor_bridge_e2e.rs | src/io/hw.rs:3
| append_flash_audit | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/
| run_flash_preflight | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por varredura estatica | src
| j1939_uds_req_id | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/io
| is_uds_response_id | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/
| recv_can_uds_response | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por varredura estatica | s
| send_uds_and_wait | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/i
| live_flash_ecm_firmware | src/io/live_runner.rs | critica | src/io/production_program.rs ; src/main.rs | src/io/live
| uds_response_id_matches_expected_target | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por varr
| uds_response_id_rejects_wrong_pf_ps_or_sa | src/io/live_runner.rs | critica | chamadas diretas nao detectadas por va
| run_all_phases | src/io/production_program.rs | critica | src/bin/simulator_cli.rs ; tests/vendor_bridge_e2e.rs | sr
| run_sim_identity_check | src/io/production_program.rs | critica | chamadas diretas nao detectadas por varredura esta
| build_conformance_summary | src/io/production_program.rs | critica | chamadas diretas nao detectadas por varredura e
| run_uds_runtime_self_test | src/io/production_program.rs | critica | chamadas diretas nao detectadas por varredura e
| supports | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/io/ven
| open | src/io/vendor_cat_comm.rs | critica | src/io/hw.rs ; src/io/hw.rs ; src/io/hw.rs ; src/io/hw.rs ; src/io/repl
| ensure_connected | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura estatica | sr
| send_bridge_request | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura estatica |
| read_bridge_response | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura estatica
| append_diag_record | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura estatica |
| init | src/io/vendor_cat_comm.rs | critica | src/io/hw.rs ; src/io/hw.rs ; src/io/live_runner.rs ; src/io/live_runne
| read_frame | src/io/vendor_cat_comm.rs | critica | src/io/hw.rs ; src/io/live_runner.rs ; src/io/live_runner.rs ; sr
| try_read_frame | src/io/vendor_cat_comm.rs | critica | src/io/hw.rs ; src/io/mock.rs ; src/io/serial_adapter.rs ; sr
| send_frame | src/io/vendor_cat_comm.rs | critica | src/io/hw.rs ; src/io/live_runner.rs ; src/io/live_runner.rs ; sr
| close | src/io/vendor_cat_comm.rs | critica | src/io/hw.rs ; src/io/hw.rs ; src/io/live_runner.rs ; src/io/live_runn
| adapter_info | src/io/vendor_cat_comm.rs | critica | src/io/hw.rs ; src/io/hw.rs | src/io/vendor_cat_comm.rs:305 ; s
| resolve_bridge_exe | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura estatica |
| map_bridge_error | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura estatica | sr
| bridge_info_from_response | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura esta
| ensure_bridge_compatibility | src/io/vendor_cat_comm.rs | critica | chamadas diretas nao detectadas por varredura es
| now_ms | src/io/vendor_cat_comm.rs | critica | src/bin/cat_comm_bridge.rs ; src/bin/cat_comm_bridge.rs ; src/io/hw.r
| engineering_oil_catalog | src/leak_physics.rs | critica | chamadas diretas nao detectadas por varredura estatica | s
| uds_process_sa | src/main.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/main.rs:6236 |
| uds_send_and_log | src/main.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/main.rs:6246
| run_uds_flash_pipeline | src/main.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/main.r
| tab_uds | src/main.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/main.rs:9040 | impact
| fmt | src/uds.rs | critica | src/adas.rs ; src/autonomous.rs ; src/autonomous.rs ; src/autonomous.rs ; src/autonomou
| new | src/uds.rs | critica | src/adas.rs ; src/autonomous.rs ; src/autonomous.rs ; src/autonomous.rs ; src/autonomou
| tick | src/uds.rs | critica | src/autonomous.rs ; src/autonomous.rs ; src/autonomous.rs ; src/autonomous.rs ; src/au
| process | src/uds.rs | critica | src/can_gateway.rs ; src/io/production_program.rs ; src/io/production_program.rs ;
| svc_session_control | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:320
| svc_ecu_reset | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:353 | imp
| svc_clear_dtc | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:395 | imp

```
| svc_read_dtc | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:428 | impa
| svc_read_data_by_id | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:504
| svc_security_access | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:554
| svc_write_data_by_id | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:64
| svc_routine_control | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:684
| svc_request_download | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:73
| svc_transfer_data | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:767 |
| svc_request_transfer_exit | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.
| svc_tester_present | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:830
| record_dtc_set | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:850 | im
| nrc | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:879 | impacto local
| log | src/uds.rs | critica | src/can_gateway.rs ; src/can_gateway.rs ; src/can_gateway.rs ; src/nvm.rs | src/uds.rs:
| service_name | src/uds.rs | critica | src/main.rs | src/uds.rs:905 ; src/main.rs:6252 | pode quebrar protocolo, gate
| dtc_to_3bytes | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | src/uds.rs:926 | imp
| session_default_relocks_security | src/uds.rs | critica | chamadas diretas nao detectadas por varredura estatica | s
| transfer_exit_rejects_incomplete_download | src/uds.rs | critica | chamadas diretas nao detectadas por varredura est
| bin_path | tests/vendor_bridge_e2e.rs | critica | chamadas diretas nao detectadas por varredura estatica | tests/ven
| unique_temp_dir | tests/vendor_bridge_e2e.rs | critica | chamadas diretas nao detectadas por varredura estatica | te
| cat_comm_bridge_probe_and_live_phases_flash_e2e | tests/vendor_bridge_e2e.rs | critica | chamadas diretas nao detect
| from_path | src/io/allowlist.rs | alta | src/io/mock.rs ; src/leak_physics.rs | src/io/allowlist.rs:33 ; src/io/mock
| is_allowed | src/io/allowlist.rs | alta | src/io/mock.rs | src/io/allowlist.rs:42 ; src/io/mock.rs:133 | pode quebra
| per_rule_rate_limit | src/io/allowlist.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io/a
| validate | src/io/allowlist.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io/allowlist.rs
| rule_matches_frame | src/io/allowlist.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io/al
| parse_u32_hex_or_dec | src/io/allowlist.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io/
| parse_mask | src/io/allowlist.rs | alta | tests/io_mock.rs | src/io/allowlist.rs:146 ; tests/io_mock.rs:190 | pode q
| match_hex_pattern | src/io/allowlist.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io/all
| can_rule_matches_mask | src/io/allowlist.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io
| serial_rule_matches_pattern | src/io/allowlist.rs | alta | chamadas diretas nao detectadas por varredura estatica |
| write_intent_enabled | src/io/hw.rs | alta | src/io/mock.rs ; tests/io_mock.rs | src/io/hw.rs:223 ; src/io/mock.rs:1
| connect_ecm | src/io/live_runner.rs | alta | src/io/production_program.rs ; src/main.rs | src/io/live_runner.rs:174
| live_preflight_ecm | src/io/live_runner.rs | alta | src/io/production_program.rs | src/io/live_runner.rs:793 ; src/i
| check_live_channel_policy | src/io/live_runner.rs | alta | chamadas diretas nao detectadas por varredura estatica |
| on_rate_limited | src/io/metrics.rs | alta | src/io/mock.rs | src/io/metrics.rs:37 ; src/io/mock.rs:149 | pode quebr
| inject_disconnect | src/io/mock.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io/mock.rs:
| clear_disconnect | src/io/mock.rs | alta | chamadas diretas nao detectadas por varredura estatica | src/io/mock.rs:3
| read_frame | src/io/mock.rs | alta | src/io/hw.rs ; src/io/live_runner.rs ; src/io/live_runner.rs ; src/io/live_runn
| try_read_frame | src/io/mock.rs | alta | src/io/hw.rs ; src/io/serial_adapter.rs ; src/io/socketcan_adapter.rs ; src
| send_frame | src/io/mock.rs | alta | src/io/hw.rs ; src/io/live_runner.rs ; src/io/live_runner.rs ; src/io/serial_ad
| default | src/io/production_program.rs | alta | src/autonomous.rs ; src/bin/cat_comm_bridge.rs ; src/bin/cat_comm_br
| run_sim_baseline_check | src/io/production_program.rs | alta | chamadas diretas nao detectadas por varredura estatic
| run_sim_connectivity_check | src/io/production_program.rs | alta | chamadas diretas nao detectadas por varredura est
... truncated 74 lines for this snapshot in markdown volume ...
```

# File Deep Dive: docs/FULL_DESIGN_CODE_BOOK.md

Role: human-facing architecture, protocols, runbooks, and design records

Line count: 36726 | Non-empty: 36352 | Symbol count: 186

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | AutoBreaking Full Design-Book and Code-Book | namespace and compile-time linkage |
| 9 | h2 | Scope | namespace and compile-time linkage |
| 16 | h2 | Build and Feature Context | namespace and compile-time linkage |
| 24 | h2 | System Call-Flow (High Level) | namespace and compile-time linkage |
| 32 | h2 | File Inventory | namespace and compile-time linkage |
| 97 | h2 | File: src/adas.rs | namespace and compile-time linkage |
| 102 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 117 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 374 | h2 | File: src/autonomous.rs | namespace and compile-time linkage |
| 379 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 421 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 1073 | h2 | File: src/bin/cat_comm_bridge.rs | namespace and compile-time linkage |
| 1078 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 1098 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 1470 | h2 | File: src/bin/simulator_cli.rs | namespace and compile-time linkage |
| 1475 | h3 | Symbol Inventory | namespace and compile-time linkage |
```

| 1484 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 1631 | h2 | File: src/boot_sequence.rs | namespace and compile-time linkage |
| 1636 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 1670 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 2192 | h2 | File: src/camera.rs | namespace and compile-time linkage |
| 2197 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 2240 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 2847 | h2 | File: src/can_bus.rs | namespace and compile-time linkage |
| 2852 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 2865 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 3050 | h2 | File: src/can_gateway.rs | namespace and compile-time linkage |
| 3055 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 3086 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 3408 | h2 | File: src/can_network.rs | namespace and compile-time linkage |
| 3413 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 3491 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 4877 | h2 | File: src/chassis.rs | namespace and compile-time linkage |
| 4882 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 4895 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 5084 | h2 | File: src/ecm_heavy.rs | namespace and compile-time linkage |
| 5089 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 5117 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 5498 | h2 | File: src/ecm_mock_provider.rs | namespace and compile-time linkage |
| 5503 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 5512 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 5715 | h2 | File: src/ecu_abs.rs | namespace and compile-time linkage |
| 5720 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 5762 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 6305 | h2 | File: src/ecu_bcm.rs | namespace and compile-time linkage |
| 6310 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 6335 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 6675 | h2 | File: src/ecu_ecm.rs | namespace and compile-time linkage |
| 6680 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 6723 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 7588 | h2 | File: src/ecu_hcm.rs | namespace and compile-time linkage |
| 7593 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 7619 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 8023 | h2 | File: src/ecu_icm.rs | namespace and compile-time linkage |
| 8028 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 8061 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 8533 | h2 | File: src/ecu_tcm.rs | namespace and compile-time linkage |
| 8538 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 8586 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 9307 | h2 | File: src/ecu_vcm.rs | namespace and compile-time linkage |
| 9312 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 9352 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 9779 | h2 | File: src/electrical.rs | namespace and compile-time linkage |
| 9784 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 9824 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 10286 | h2 | File: src/engine.rs | namespace and compile-time linkage |
| 10291 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 10306 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 10437 | h2 | File: src/gps.rs | namespace and compile-time linkage |
| 10442 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 10467 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 10839 | h2 | File: src/implement.rs | namespace and compile-time linkage |
| 10844 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 10885 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 11395 | h2 | File: src/imu.rs | namespace and compile-time linkage |
| 11400 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 11425 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 11777 | h2 | File: src/io/allowlist.rs | namespace and compile-time linkage |
| 11782 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 11802 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 12037 | h2 | File: src/io/artifact.rs | namespace and compile-time linkage |
| 12042 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 12049 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 12107 | h2 | File: src/io/ecm_params.rs | namespace and compile-time linkage |
| 12112 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 12128 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 12290 | h2 | File: src/io/hw.rs | namespace and compile-time linkage |
| 12295 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 12329 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 12739 | h2 | File: src/io/live_runner.rs | namespace and compile-time linkage |
| 12744 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 12814 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |

| 14495 | h2 | File: src/io/metrics.rs | namespace and compile-time linkage |
| 14500 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 14515 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 14568 | h2 | File: src/io/mock.rs | namespace and compile-time linkage |
| 14573 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 14591 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 14783 | h2 | File: src/io/mod.rs | namespace and compile-time linkage |
| 14788 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 14806 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 14826 | h2 | File: src/io/production_program.rs | namespace and compile-time linkage |
| 14831 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 14854 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 15656 | h2 | File: src/io/rate_limiter.rs | namespace and compile-time linkage |
| 15661 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 15678 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 15768 | h2 | File: src/io/replay.rs | namespace and compile-time linkage |
| 15773 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 15789 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 15977 | h2 | File: src/io/serial_adapter.rs | namespace and compile-time linkage |
| 15982 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 15997 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 16098 | h2 | File: src/io/socketcan_adapter.rs | namespace and compile-time linkage |
| 16103 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 16121 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 16271 | h2 | File: src/io/vendor_cat_comm.rs | namespace and compile-time linkage |
| 16276 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 16308 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 16731 | h2 | File: src/j1939.rs | namespace and compile-time linkage |
| 16736 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 16870 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 18531 | h2 | File: src/leak_physics.rs | namespace and compile-time linkage |
| 18536 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 18636 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 20792 | h2 | File: src/lib.rs | namespace and compile-time linkage |
| 20797 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 20879 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 22053 | h2 | File: src/lidar.rs | namespace and compile-time linkage |
| 22058 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 22081 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 22442 | h2 | File: src/main.rs | namespace and compile-time linkage |
| 22447 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 22565 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 32036 | h2 | File: src/network_mgmt.rs | namespace and compile-time linkage |
| 32041 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 32069 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 32414 | h2 | File: src/nvm.rs | namespace and compile-time linkage |
| 32419 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 32464 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 32854 | h2 | File: src/observability.rs | namespace and compile-time linkage |
| 32859 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 32871 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 32927 | h2 | File: src/radar.rs | namespace and compile-time linkage |
| 32932 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 32962 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 33319 | h2 | File: src/sensors.rs | namespace and compile-time linkage |
| 33324 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 33335 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 33508 | h2 | File: src/sim_core.rs | namespace and compile-time linkage |
| 33513 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 33540 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 33616 | h2 | File: src/transmission.rs | namespace and compile-time linkage |
| 33621 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 33641 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 33787 | h2 | File: src/uds.rs | namespace and compile-time linkage |
| 33792 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 33869 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 34836 | h2 | File: src/v2x_telematics.rs | namespace and compile-time linkage |
| 34841 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 34880 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 35519 | h2 | File: src/vehicle.rs | namespace and compile-time linkage |
| 35524 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 35544 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 35697 | h2 | File: src/widgets.rs | namespace and compile-time linkage |
| 35702 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 35715 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 36047 | h2 | File: tests/io_mock.rs | namespace and compile-time linkage |

```
| 36052 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 36066 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 36279 | h2 | File: tests/leak_system_integration.rs | namespace and compile-time linkage |
| 36284 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 36291 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 36380 | h2 | File: tests/property_invariants.rs | namespace and compile-time linkage |
| 36385 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 36392 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 36435 | h2 | File: tests/speed_regression.rs | namespace and compile-time linkage |
| 36440 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 36449 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 36520 | h2 | File: tests/system_failure_scenarios.rs | namespace and compile-time linkage |
| 36525 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 36536 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 36622 | h2 | File: tests/vendor_bridge_e2e.rs | namespace and compile-time linkage |
| 36627 | h3 | Symbol Inventory | namespace and compile-time linkage |
| 36635 | h3 | Line-by-Line Impact Map | namespace and compile-time linkage |
| 36721 | h2 | Change Safety Guide | namespace and compile-time linkage |
```

## Content Snapshot

```
# AutoBreaking Full Design-Book and Code-Book

> Generated automatically from the current repository snapshot.
> Goal: provide deep technical traceability with architecture, call-flow links, symbol map, and line-level impact note


Generated at: **2026-08-13 09:38:48**

## Scope

- Full repository code map for Rust source and tests.
- Architecture and runtime flow linking major modules.
- Per-file symbol inventory with line references.
- Line-by-line impact commentary for change-risk analysis.

## Build and Feature Context

- name = "auto_breaking"
- version = "0.3.0"
- edition = "2021"
- name = "solver_bench"
- Key features: default, advanced_observability, new_integrator, vendor-windows.

## System Call-Flow (High Level)

1. UI path: src/main.rs drives simulation ticks, visualizations, and control state updates.
2. CLI path: src/bin/simulator_cli.rs parses hardware/phase flags and calls io::production_program::run_all_phases.
3. Live flash path: io::production_program -> io::live_runner -> io::hw selected adapter -> io::vendor_cat_comm or soc
4. UDS protocol path: io::live_runner and uds.rs coordinate 0x27/0x34/0x36/0x37 services, transport accounting, and co
5. Report path: production report JSON/Markdown emitted with gate and conformance summary.

## File Inventory

Total files covered: **60**

- src/adas.rs
- src/autonomous.rs
- src/bin/cat_comm_bridge.rs
- src/bin/simulator_cli.rs
- src/boot_sequence.rs
- src/camera.rs
- src/can_bus.rs
- src/can_gateway.rs
- src/can_network.rs
- src/chassis.rs
- src/ecm_heavy.rs
- src/ecm_mock_provider.rs
- src/ecu_abs.rs
- src/ecu_bcm.rs
- src/ecu_ecm.rs
- src/ecu_hcm.rs
```

```
- src/ecu_icm.rs
- src/ecu_tcm.rs
- src/ecu_vcm.rs
- src/electrical.rs
- src/engine.rs
- src/gps.rs
- src/implement.rs
- src/imu.rs
- src/io/allowlist.rs
- src/io/artifact.rs
- src/io/ecm_params.rs
- src/io/hw.rs
- src/io/live_runner.rs
- src/io/metrics.rs
- src/io/mock.rs
- src/io/mod.rs
- src/io/production_program.rs
- src/io/rate_limiter.rs
- src/io/replay.rs
- src/io/serial_adapter.rs
- src/io/socketcan_adapter.rs
- src/io/vendor_cat_comm.rs
- src/j1939.rs
- src/leak_physics.rs
- src/lib.rs
- src/lidar.rs
- src/main.rs
- src/network_mgmt.rs
- src/nvm.rs
- src/observability.rs
- src/radar.rs
- src/sensors.rs
- src/sim_core.rs
- src/transmission.rs
- src/uds.rs
- src/v2x_telematics.rs
- src/vehicle.rs
- src/widgets.rs
- tests/io_mock.rs
- tests/leak_system_integration.rs
- tests/property_invariants.rs
- tests/speed_regression.rs
- tests/system_failure_scenarios.rs
- tests/vendor_bridge_e2e.rs

## File: src/adas.rs

Role: Domain module in simulation/control/protocol stack.
Lines: 252 | Non-empty: 219 | Symbols: 10

### Symbol Inventory

| Line | Kind | Symbol | Change Impact |
|---:|---|---|---|
| 5 | struct | AdasModule | data/structure coupling |
| 48 | enum | ParkZone | data/structure coupling |
| 55 | impl | std::fmt::Display for ParkZone | behavior binding and method semantics |
| 56 | fn | fmt | signature/API coupling |
| 66 | impl | AdasModule | behavior binding and method semantics |
| 67 | fn | new | signature/API coupling |
| 105 | fn | update | signature/API coupling |
| 238 | fn | toggle_acc | signature/API coupling |
| 246 | fn | acc_speed_up | signature/API coupling |
| 249 | fn | acc_speed_down | signature/API coupling |

### Line-by-Line Impact Map

| Line | Source | Operational Impact If Changed |
|---:|---|---|
| 1 | use crate::sensors::SensorSuite; | imports symbols; changing path/name affects compile-time resolution. |
| 2 | (blank) | line is blank; visual separation only. |
```

```
| 3 | /// ADAS domain controller (SAE Level 2) | comment line; documentation intent, no runtime effect. |
| 4 | #[derive(Debug, Clone)] | executable/support line; behavior impact depends on surrounding block context. |
| 5 | pub struct AdasModule { | declares data layout; field changes may break serialization, tests, and callers. |
| 6 |     // Adaptive Cruise Control | comment line; documentation intent, no runtime effect. |
| 7 |     pub acc_enabled: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 8 |     pub acc_set_speed: f64,    // km/h | executable/support line; behavior impact depends on surrounding block c
| 9 |     pub acc_headway_time: f64, // s (gap to lead) | executable/support line; behavior impact depends on surround
| 10 |     pub acc_throttle_out: f64, | executable/support line; behavior impact depends on surrounding block context.
| 11 |     pub acc_brake_out: f64, | executable/support line; behavior impact depends on surrounding block context. |
| 12 |     acc_speed_error_integral: f64, | executable/support line; behavior impact depends on surrounding block cont
| 13 | (blank) | line is blank; visual separation only. |
| 14 |     // Autonomous Emergency Braking | comment line; documentation intent, no runtime effect. |
| 15 |     pub aeb_enabled: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 16 |     pub aeb_pre_charge: bool, // brakes pre-charged | executable/support line; behavior impact depends on surro
| 17 |     pub aeb_active: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 18 |     pub aeb_ttc: f64, // seconds to collision | executable/support line; behavior impact depends on surrounding
| 19 | (blank) | line is blank; visual separation only. |
| 20 |     // Lane Keeping Assist / Lane Departure Warning | comment line; documentation intent, no runtime effect. |
| 21 |     pub lka_enabled: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 22 |     pub lka_active: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 23 |     pub lka_steering_correction: f64, | executable/support line; behavior impact depends on surrounding block c
| 24 |     pub ldw_warning: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 25 | (blank) | line is blank; visual separation only. |
| 26 |     // Blind Spot Monitoring | comment line; documentation intent, no runtime effect. |
| 27 |     pub bsm_enabled: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 28 |     pub bsm_left_warning: bool, | executable/support line; behavior impact depends on surrounding block context
| 29 |     pub bsm_right_warning: bool, | executable/support line; behavior impact depends on surrounding block contex
| 30 | (blank) | line is blank; visual separation only. |
| 31 |     // Park Assist (front/rear ultrasonic guidance) | comment line; documentation intent, no runtime effect. |
| 32 |     pub park_assist_enabled: bool, | executable/support line; behavior impact depends on surrounding block cont
| 33 |     pub park_assist_zone: ParkZone, | executable/support line; behavior impact depends on surrounding block con
| 34 | (blank) | line is blank; visual separation only. |
| 35 |     // Traffic Sign Recognition | comment line; documentation intent, no runtime effect. |
| 36 |     pub tsr_active: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 37 |     pub tsr_speed_limit: Option<u32>, | executable/support line; behavior impact depends on surrounding block c
| 38 | (blank) | line is blank; visual separation only. |
| 39 |     // Driver Attention Monitor (simulated) | comment line; documentation intent, no runtime effect. |
| 40 |     pub dam_drowsiness: f64, // 0-1 | executable/support line; behavior impact depends on surrounding block con
| 41 |     pub dam_alert: bool, | executable/support line; behavior impact depends on surrounding block context. |
| 42 | (blank) | line is blank; visual separation only. |
| 43 |     // Autonomous level: 0=none, 1=partial, 2=combined | comment line; documentation intent, no runtime effect.
| 44 |     pub autonomy_level: u8, | executable/support line; behavior impact depends on surrounding block context. |
| 45 | } | scope delimiter; structure-only line, but misplaced edits can change ownership and lifetime scopes. |
| 46 | (blank) | line is blank; visual separation only. |
| 47 | #[derive(Debug, Clone, Copy, PartialEq)] | executable/support line; behavior impact depends on surrounding bloc
| 48 | pub enum ParkZone { | declares state variants; adding/removing variants impacts pattern matches and logic branc
| 49 |     Clear, | executable/support line; behavior impact depends on surrounding block context. |
| 50 |     Caution, | executable/support line; behavior impact depends on surrounding block context. |
| 51 |     Warning, | executable/support line; behavior impact depends on surrounding block context. |
| 52 |     Critical, | executable/support line; behavior impact depends on surrounding block context. |
| 53 | } | scope delimiter; structure-only line, but misplaced edits can change ownership and lifetime scopes. |
| 54 | (blank) | line is blank; visual separation only. |
| 55 | impl std::fmt::Display for ParkZone { | implementation block; method behavior and trait conformance are defined
| 56 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | declares executable behavior; signat
| 57 |         match self { | branch dispatch; changing arms alters behavior for variants and input classes. |
| 58 |             ParkZone::Clear => write!(f, "CLEAR"), | I/O or protocol-critical logic; edits can affect integrati
| 59 |             ParkZone::Caution => write!(f, "CAUTION"), | I/O or protocol-critical logic; edits can affect integ
| 60 |             ParkZone::Warning => write!(f, "WARNING"), | I/O or protocol-critical logic; edits can affect integ
| 61 |             ParkZone::Critical => write!(f, "CRITICAL"), | I/O or protocol-critical logic; edits can affect int
| 62 |         } | scope delimiter; structure-only line, but misplaced edits can change ownership and lifetime scopes.
| 63 |     } | scope delimiter; structure-only line, but misplaced edits can change ownership and lifetime scopes. |
| 64 | } | scope delimiter; structure-only line, but misplaced edits can change ownership and lifetime scopes. |
| 65 | (blank) | line is blank; visual separation only. |
| 66 | impl AdasModule { | implementation block; method behavior and trait conformance are defined here. |
| 67 |     pub fn new() -> Self { | declares executable behavior; signature/body changes alter API, control flow, and
| 68 |         Self { | executable/support line; behavior impact depends on surrounding block context. |
| 69 |             acc_enabled: false, | executable/support line; behavior impact depends on surrounding block context
| 70 |             acc_set_speed: 80.0, | executable/support line; behavior impact depends on surrounding block contex
| 71 |             acc_headway_time: 2.0, | executable/support line; behavior impact depends on surrounding block cont
| 72 |             acc_throttle_out: 0.0, | executable/support line; behavior impact depends on surrounding block cont
| 73 |             acc_brake_out: 0.0, | executable/support line; behavior impact depends on surrounding block context
```

```
| 74 |             acc_speed_error_integral: 0.0, | executable/support line; behavior impact depends on surrounding bl
| 75 | (blank) | line is blank; visual separation only. |
| 76 |             aeb_enabled: true, | executable/support line; behavior impact depends on surrounding block context.
| 77 |             aeb_pre_charge: false, | executable/support line; behavior impact depends on surrounding block cont
| 78 |             aeb_active: false, | executable/support line; behavior impact depends on surrounding block context.
| 79 |             aeb_ttc: f64::INFINITY, | executable/support line; behavior impact depends on surrounding block con
| 80 | (blank) | line is blank; visual separation only. |
| 81 |             lka_enabled: false, | executable/support line; behavior impact depends on surrounding block context
| 82 |             lka_active: false, | executable/support line; behavior impact depends on surrounding block context.
| 83 |             lka_steering_correction: 0.0, | executable/support line; behavior impact depends on surrounding blo
| 84 |             ldw_warning: false, | executable/support line; behavior impact depends on surrounding block context
| 85 | (blank) | line is blank; visual separation only. |
| 86 |             bsm_enabled: true, | executable/support line; behavior impact depends on surrounding block context.
| 87 |             bsm_left_warning: false, | executable/support line; behavior impact depends on surrounding block co
| 88 |             bsm_right_warning: false, | executable/support line; behavior impact depends on surrounding block c
| 89 | (blank) | line is blank; visual separation only. |
| 90 |             park_assist_enabled: false, | executable/support line; behavior impact depends on surrounding block
| 91 |             park_assist_zone: ParkZone::Clear, | executable/support line; behavior impact depends on surroundin
| 92 | (blank) | line is blank; visual separation only. |
| 93 |             tsr_active: true, | executable/support line; behavior impact depends on surrounding block context.
| 94 |             tsr_speed_limit: Some(80), | executable/support line; behavior impact depends on surrounding block
| 95 | (blank) | line is blank; visual separation only. |
| 96 |             dam_drowsiness: 0.0, | executable/support line; behavior impact depends on surrounding block contex
| 97 |             dam_alert: false, | executable/support line; behavior impact depends on surrounding block context.
| 98 | (blank) | line is blank; visual separation only. |
| 99 |             autonomy_level: 2, | executable/support line; behavior impact depends on surrounding block context.
| 100 |         } | scope delimiter; structure-only line, but misplaced edits can change ownership and lifetime scopes
... truncated 36506 lines for this snapshot in markdown volume ...
```

# File Deep Dive: docs/OSS_INTEGRATION_MATRIX.md

Role: human-facing architecture, protocols, runbooks, and design records

Line count: 160 | Non-empty: 113 | Symbol count: 17

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | OSS Integration Matrix for AutoBreaking | namespace and compile-time linkage |
| 3 | h2 | Objective | namespace and compile-time linkage |
| 14 | h2 | Selection Rules (hard gates) | namespace and compile-time linkage |
| 22 | h2 | Candidate Matrix | namespace and compile-time linkage |
| 34 | h2 | Module-to-OSS Mapping | namespace and compile-time linkage |
| 36 | h2 | src/io/live_runner.rs | namespace and compile-time linkage |
| 55 | h2 | src/io/vendor_cat_comm.rs | namespace and compile-time linkage |
| 70 | h2 | src/io/production_program.rs | namespace and compile-time linkage |
| 83 | h2 | src/io/hw.rs | namespace and compile-time linkage |
| 100 | h2 | License and Compliance Notes | namespace and compile-time linkage |
| 113 | h2 | Priority Adoption Plan | namespace and compile-time linkage |
| 115 | h2 | Wave 1 (immediate, low risk) | namespace and compile-time linkage |
| 121 | h2 | Wave 2 (short-term) | namespace and compile-time linkage |
| 127 | h2 | Wave 3 (optional) | namespace and compile-time linkage |
| 132 | h2 | Concrete Backlog Items | namespace and compile-time linkage |
| 145 | h2 | Verification Gates for First Integration Patch Set | namespace and compile-time linkage |
| 152 | h2 | External References (evaluated) | namespace and compile-time linkage |
```

## Content Snapshot

```
# OSS Integration Matrix for AutoBreaking

## Objective

Turn verified open-source protocol stacks and tools into production-ready integration targets for the current AutoBrea

Scope mapped to current modules:

- src/io/live_runner.rs
- src/io/vendor_cat_comm.rs
- src/io/production_program.rs
```

```
- src/io/hw.rs

## Selection Rules (hard gates)

1. License must be compatible with commercial/distribution constraints.
2. Stack must expose explicit behavior for UDS 0x27/0x34/0x36/0x37 or ISO-TP flow control semantics.
3. Integration must preserve existing write gates:
   - enable_write + allowlist + noninteractive_approved + !dry_run.
4. No regression on current strict phase runner behavior.

## Candidate Matrix

| Candidate | Type | License | Maturity | Protocol coverage | OS fit | Integration effort | Main risks | Recommendatio
|---|---|---|---|---|---|---|---|---|
| driftregion/iso14229 | C library | MIT | Medium/High | UDS server/client core, 0x27, 0x34, 0x36, 0x37 semantics | Cr
| linux-can/can-utils | Linux CLI/tools + headers | Mixed permissive/GPL dual in many files | High | CAN + ISO-TP oper
| pylessard/python-can-isotp | Python ISO-TP stack | MIT | High | ISO-TP timing, FC blocksize/stmin/wftmax, error hand
| pylessard/python-udsoncan | Python UDS client/services | MIT | High | UDS services and strict response handling, inc
| socketcan-rs/socketcan-rs | Rust crate | MIT/Apache-2.0 | Medium/High | SocketCAN read/write abstractions (CAN path)
| hardbyte/python-can | Python CAN abstraction | LGPL-3.0 | High | Multi-adapter CAN APIs, filtering models, send/recv
| cantools/cantools | Python DBC tooling | MIT | High | DBC decode/encode and message schema workflows | Cross-platfor

## Module-to-OSS Mapping

## src/io/live_runner.rs

Use OSS references to tighten transport and diagnostic semantics:

- ISO-TP behavior parity:
  - fc stmin, blocksize, wftmax handling patterns from python-can-isotp.
  - timeout taxonomy parity for flow control and consecutive frame windows.
- UDS transaction strictness:
  - 0x27 seed/key sequence controls from iso14229 + udsoncan behavior.
  - 0x34/0x36/0x37 state machine and NRC gate rules.
- J1939/CAN observability:
  - can-utils style operational traces as external validation during integration tests.

Targeted outcomes:

1. Deterministic block counter and sequence checks during transfer.
2. Explicit FC timeout and wrong-sequence failure categories in report evidence.
3. Stronger parity between preflight and flash-state transitions.

## src/io/vendor_cat_comm.rs

Use OSS as protocol discipline references for the template bridge:

- Borrow transport option vocabulary from ISO-TP tool ecosystem:
  - stmin, bs, wftmax, padding mode, tx/rx separation semantics.
- Add bridge capability negotiation concept:
  - version and transport capability fields inspired by robust adapter APIs.

Targeted outcomes:

1. Bridge init handshake returns capabilities and limits.
2. Host applies strict compatibility checks before enabling flash path.
3. Better error kind granularity mapping to existing HwError categories.

## src/io/production_program.rs

Use OSS conformance references to harden phase evidence:

- Add conformance checks for UDS service outcomes (0x27/0x34/0x36/0x37).
- Add explicit blocked/failed distinction for transport-timeout vs policy-gate failures.

Targeted outcomes:

1. Per-phase evidence includes protocol reason class.
2. JSON report includes conformance_summary section.
3. Strict mode fails on semantic mismatch, not only transport failure.
## src/io/hw.rs
```

```
Use OSS coverage findings to guide adapter capability model:

- Define capability flags:
  - can_raw
  - isotp
  - j1939
  - uds_flash
  - vendor_bridge
- Drive startup policy and phase gates by declared capability profile.

Targeted outcomes:

1. Predictable failures when required capability is absent.
2. Reduced false-start runs in production phases.

## License and Compliance Notes

Do:

1. Treat can-utils and other GPL-affected projects as interoperability references and external tools.
2. Keep source-level implementation in Rust under this repository's own licensing strategy.
3. Keep copied material to zero; only behavior-level alignment and independent implementation.

Do not:

1. Copy/paste implementation code from GPL files into this codebase.
2. Introduce Python runtime as a hard dependency in production path unless approved.

## Priority Adoption Plan

## Wave 1 (immediate, low risk)

1. Add protocol conformance checklist to phase report generation in production_program.
2. Add ISO-TP timing and FC diagnostics fields to live_runner flash summary.
3. Add vendor bridge capability negotiation fields in vendor_cat_comm init/ping flow.

## Wave 2 (short-term)

1. Introduce adapter capability struct in hw and enforce it in phase gates.
2. Add repeatable interoperability scripts using can-utils on Linux benches.
3. Create golden UDS/ISO-TP test vectors from OSS behavior references.

## Wave 3 (optional)

1. Evaluate socketcan-rs deeper integration for Linux adapter internals.
2. Add offline DBC validation workflow using cantools-equivalent process (tooling side).

## Concrete Backlog Items

1. live_runner: add FlashTransportDiagnostics struct:
   - fc_blocksize_seen
   - fc_stmin_seen
   - wait_frame_count
   - sequence_error_count
   - flowcontrol_timeout_count
2. production_program: add conformance_summary in ProgramReport with per-service pass/fail.
3. hw: add capability declaration and gate checks before live flash preflight.
4. vendor_cat_comm: extend bridge protocol with protocol_version and capability list.
5. docs: append runbook section for Linux can-utils validation workflow.

## Verification Gates for First Integration Patch Set

1. cargo check must pass.
2. cargo test --workspace must pass.
3. Existing strict phase CLI run must still produce overall_passed=true in baseline path.
4. New report fields must be present and non-empty when flash path runs.

## External References (evaluated)

- https://github.com/driftregion/iso14229
- https://github.com/linux-can/can-utils
```

```
- https://github.com/pylessard/python-can-isotp
- https://github.com/pylessard/python-udsoncan
- https://github.com/socketcan-rs/socketcan-rs
- https://github.com/hardbyte/python-can
- https://github.com/cantools/cantools
```

# File Deep Dive: fuzz/Cargo.toml

Role: project support artifact

Line count: 19 | Non-empty: 16 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | [package] | support logic; impact depends on neighboring block and data flow. |
| 2 | name = "auto_breaking-fuzz" | support logic; impact depends on neighboring block and data flow. |
| 3 | version = "0.0.1" | support logic; impact depends on neighboring block and data flow. |
| 4 | edition = "2021" | support logic; impact depends on neighboring block and data flow. |
| 5 | publish = false | support logic; impact depends on neighboring block and data flow. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | [package.metadata] | support logic; impact depends on neighboring block and data flow. |
| 8 | cargo-fuzz = true | support logic; impact depends on neighboring block and data flow. |
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | [dependencies] | support logic; impact depends on neighboring block and data flow. |
| 11 | libfuzzer-sys = "0.4" | support logic; impact depends on neighboring block and data flow. |
| 12 | auto_breaking = { path = ".." } | support logic; impact depends on neighboring block and data flow. |
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 | [[bin]] | support logic; impact depends on neighboring block and data flow. |
| 15 | name = "j1939_frame_from_raw" | support logic; impact depends on neighboring block and data flow. |
| 16 | path = "fuzz_targets/j1939_frame_from_raw.rs" | support logic; impact depends on neighboring block and data flow
| 17 | test = false | support logic; impact depends on neighboring block and data flow. |
| 18 | doc = false | support logic; impact depends on neighboring block and data flow. |
| 19 | bench = false | support logic; impact depends on neighboring block and data flow. |
```

# File Deep Dive: fuzz/fuzz_targets/j1939_frame_from_raw.rs

Role: project support artifact

Line count: 16 | Non-empty: 13 | Symbol count: 0

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| len | 2 |
| from_raw | 1 |
| pgn_name | 1 |
| sa_name | 1 |
| data_hex | 1 |
```

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #![no_main] | comment/documentation; changing affects understanding and intent traceability. |
| 2 | use libfuzzer_sys::fuzz_target; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | fuzz_target!(\|data: &[u8]\| { | support logic; impact depends on neighboring block and data flow. |
```

```
| 5 |     if data.len() < 4 { | runtime gate; condition changes can enable/disable critical paths. |
| 6 |         return; | support logic; impact depends on neighboring block and data flow. |
| 7 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 |     let raw_id = u32::from_le_bytes([data[0], data[1], data[2], data[3]]); | support logic; impact depends on neig
| 10 |     let payload = if data.len() > 4 { &data[4..] } else { &[] }; | support logic; impact depends on neighboring b
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 |     let frame = auto_breaking::j1939::J1939Frame::from_raw(0.0, raw_id, payload); | support logic; impact depends
| 13 |     let _ = frame.pgn_name(); | support logic; impact depends on neighboring block and data flow. |
| 14 |     let _ = frame.sa_name(); | support logic; impact depends on neighboring block and data flow. |
| 15 |     let _ = frame.data_hex(); | support logic; impact depends on neighboring block and data flow. |
| 16 | }); | support logic; impact depends on neighboring block and data flow. |
```

## File Deep Dive: reports/leak_predictions_20260812_095534.json

Role: project support artifact

Line count: 3242 | Non-empty: 3242 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Content Snapshot


```
[
  {
    "circuit_name": "AC_HIGH",
    "scenario_name": "MC_RUN_0055",
    "peak_pressure_bar": 16.446337895824996,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.07078010452282983,
    "hours_to_rupture": 95.55509363296048,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "AC_HIGH",
    "scenario_name": "MC_RUN_0110",
    "peak_pressure_bar": 16.392675812141626,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.06873197544163061,
    "hours_to_rupture": 95.65732483253994,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "AC_HIGH",
    "scenario_name": "MC_RUN_0021",
    "peak_pressure_bar": 16.359510820432845,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.06748675949335486,
    "hours_to_rupture": 95.72058075556667,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "AC_HIGH",
    "scenario_name": "MC_RUN_0076",
    "peak_pressure_bar": 16.305848736749432,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.06550563052641262,
    "hours_to_rupture": 95.82305021135154,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "AC_HIGH",
    "scenario_name": "MC_RUN_0042",
    "peak_pressure_bar": 16.219021661357328,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.06238958388279249,
    "hours_to_rupture": 95.9891620156419,
    "likely_failure_mode": "MonteCarlo degradation"
```

```
},
{
  "circuit_name": "AC_HIGH",
  "scenario_name": "MC_RUN_0097",
  "peak_pressure_bar": 16.16535957767396,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.06052007114995274,
  "hours_to_rupture": 96.09201846621298,
  "likely_failure_mode": "MonteCarlo degradation"
},
{
  "circuit_name": "HYD_MAIN",
  "scenario_name": "MC_RUN_0055",
  "peak_pressure_bar": 185.0213013280312,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.059701465286342376,
  "hours_to_rupture": 55.459601962295665,
  "likely_failure_mode": "MonteCarlo degradation"
},
{
  "circuit_name": "HYD_MAIN",
  "scenario_name": "MC_RUN_0110",
  "peak_pressure_bar": 184.4176028865933,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.05958937840531402,
  "hours_to_rupture": 55.564267935768214,
  "likely_failure_mode": "MonteCarlo degradation"
},
{
  "circuit_name": "HYD_MAIN",
  "scenario_name": "MC_RUN_0021",
  "peak_pressure_bar": 184.04449672986954,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.059520169561057316,
  "hours_to_rupture": 55.62909153786054,
  "likely_failure_mode": "MonteCarlo degradation"
},
{
  "circuit_name": "HYD_MAIN",
  "scenario_name": "MC_RUN_0076",
  "peak_pressure_bar": 183.4407982884311,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.059408291871219605,
  "hours_to_rupture": 55.73419979694178,
  "likely_failure_mode": "MonteCarlo degradation"
},
{
  "circuit_name": "AC_HIGH",
  "scenario_name": "MC_RUN_0008",
  "peak_pressure_bar": 16.132194585965145,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.05938653323861292,
  "hours_to_rupture": 96.15566138470533,
  "likely_failure_mode": "MonteCarlo degradation"
},
{
  "circuit_name": "HYD_MAIN",
  "scenario_name": "MC_RUN_0042",
  "peak_pressure_bar": 182.46399369026994,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.05922754358849149,
  "hours_to_rupture": 55.904850456585756,
  "likely_failure_mode": "MonteCarlo degradation"
},
{
  "circuit_name": "HYD_MAIN",
  "scenario_name": "MC_RUN_0097",
  "peak_pressure_bar": 181.86029524883205,
  "final_alert": "Normal",
  "final_rupture_probability_pct": 0.059116004009669355,
  "hours_to_rupture": 56.0106794797478,
```

```
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0008",
    "peak_pressure_bar": 181.48718909210788,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.05904713327642971,
    "hours_to_rupture": 56.076223850347944,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0063",
    "peak_pressure_bar": 180.88349065066984,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.058935802434617025,
    "hours_to_rupture": 56.18250152720561,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0118",
    "peak_pressure_bar": 180.27979220923194,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.05882460051262949,
    "hours_to_rupture": 56.28905771736561,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0029",
    "peak_pressure_bar": 179.90668605250818,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.058755938376048776,
    "hours_to_rupture": 56.3550527333476,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0084",
    "peak_pressure_bar": 179.3029876110708,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.05864494491037051,
    "hours_to_rupture": 56.4620615695764,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0050",
    "peak_pressure_bar": 178.3261830129086,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.05846562637041467,
    "hours_to_rupture": 56.635801075746606,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0105",
    "peak_pressure_bar": 177.7224845714707,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.05835496982668875,
    "hours_to_rupture": 56.74354758867488,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0016",
    "peak_pressure_bar": 177.34937841474667,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.05828664461206183,
```

```
    "hours_to_rupture": 56.810280285937644,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0071",
    "peak_pressure_bar": 176.74567997330848,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.05817619607046364,
    "hours_to_rupture": 56.91848599815104,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0037",
    "peak_pressure_bar": 175.76887537514682,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.057997758641659465,
    "hours_to_rupture": 57.09417072331493,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0092",
    "peak_pressure_bar": 175.16517693370946,
    "final_alert": "Normal",
    "final_rupture_probability_pct": 0.057887646287132055,
    "hours_to_rupture": 57.2031248102584,
    "likely_failure_mode": "MonteCarlo degradation"
  },
  {
    "circuit_name": "HYD_MAIN",
    "scenario_name": "MC_RUN_0003",
... truncated 3022 lines for this snapshot in markdown volume ...
```


## File Deep Dive: reports/leak_report_20260812_095535.json

Role: project support artifact

Line count: 41 | Non-empty: 41 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Content Snapshot


```
[
  {
    "name": "HYD_MAIN",
    "alert": "Normal",
    "rupture_probability_pct": 0.0015014788080606474,
    "predicted_hours_to_rupture": 113.7150001905555,
    "leak_area_mm2": 0.006770400890085763,
    "leak_lpm": 0.05177865846094559,
    "pressure_min_bar": 0.0,
    "pressure_mean_bar": 173.68924582409122,
    "pressure_max_bar": 182.7986778489556,
    "rupture_confirmed": false,
    "warning_text": "O-ring circuito hidraulico principal: comportamento nominal"
  },
  {
    "name": "ENG_OIL",
    "alert": "Normal",
    "rupture_probability_pct": 0.002464149334345401,
    "predicted_hours_to_rupture": 86.00311188156583,
    "leak_area_mm2": 0.0007257605953964654,
    "leak_lpm": 0.0006042126823019544,
```

```
    "pressure_min_bar": 0.0,
    "pressure_mean_bar": 2.5181818755848355,
    "pressure_max_bar": 3.2,
    "rupture_confirmed": false,
    "warning_text": "O-ring pressao de oleo do motor: comportamento nominal"
  },
  {
    "name": "AC_HIGH",
    "alert": "Normal",
    "rupture_probability_pct": 0.0025704298332136387,
    "predicted_hours_to_rupture": 83.20107399753688,
    "leak_area_mm2": 0.003492781295727591,
    "leak_lpm": 0.0038323938516853988,
    "pressure_min_bar": 5.5,
    "pressure_mean_bar": 5.500000003338023,
    "pressure_max_bar": 12.0,
    "rupture_confirmed": false,
    "warning_text": "O-ring alta do ar condicionado: comportamento nominal"
  }
]
```


## File Deep Dive: reports/production_phase_report_1786621709828.json

Role: project support artifact

Line count: 84 | Non-empty: 84 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Content Snapshot


```
{
  "generated_at_ms": 1786621709828,
  "mode": "sim",
  "target_sa": null,
  "artifact": null,
  "detect": null,
  "preflight": null,
  "identity": null,
  "phases": [
    {
      "phase": 1,
      "title": "Baseline e Compliance",
      "status": "Passed",
      "details": "runbook e protocolo presentes"
    },
    {
      "phase": 2,
      "title": "Plataforma Fisica Real",
      "status": "Blocked",
      "details": "requer --hw-mode=live"
    },
    {
      "phase": 3,
      "title": "Conexao Confiavel",
      "status": "Blocked",
      "details": "hardware live indisponivel"
    },
    {
      "phase": 4,
      "title": "Leitura e Identidade ECM",
      "status": "Blocked",
      "details": "requer modo live"
    },
    {
      "phase": 5,
```

```
      "title": "Validacao de Artefato",
      "status": "Blocked",
      "details": "forneca --firmware=<arquivo.bin>"
    },
    {
      "phase": 6,
      "title": "Flash Seguro (Preflight)",
      "status": "Blocked",
      "details": "requer modo live"
    },
    {
      "phase": 7,
      "title": "Diagnostico e Debug",
      "status": "Passed",
      "details": "uds_retry=3 p2=1000 p2*= 5000"
    },
    {
      "phase": 8,
      "title": "Governanca e Seguranca",
      "status": "Failed",
      "details": "allowlist_present=false noninteractive_approved=false"
    },
    {
      "phase": 9,
      "title": "Confiabilidade e Stress",
      "status": "Passed",
      "details": "rate_global=100 rate_per_id=10 reconnect=500..30000 ms"
    },
    {
      "phase": 10,
      "title": "Industrializacao",
      "status": "Passed",
      "details": "pipeline de build/test encontrado"
    },
    {
      "phase": 11,
      "title": "Homologacao Final",
      "status": "Passed",
      "details": "dossie json/markdown sera gerado"
    },
    {
      "phase": 12,
      "title": "Gate de Producao",
      "status": "Failed",
      "details": "ha falhas bloqueantes nas fases anteriores"
    }
  ],
  "overall_passed": false
}
```


## File Deep Dive: reports/production_phase_report_1786621709828.md

Role: project support artifact

Line count: 21 | Non-empty: 18 | Symbol count: 2

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Production Program Report | namespace and compile-time linkage |
| 8 | h2 | Phase Results | namespace and compile-time linkage |
```

### Content Snapshot

```
# Production Program Report

- generated_at_ms: 1786621709828
- mode: sim
- target_sa: None
```

```
- overall_passed: false

## Phase Results

- Fase 1 - Baseline e Compliance: Passed (runbook e protocolo presentes)
- Fase 2 - Plataforma Fisica Real: Blocked (requer --hw-mode=live)
- Fase 3 - Conexao Confiavel: Blocked (hardware live indisponivel)
- Fase 4 - Leitura e Identidade ECM: Blocked (requer modo live)
- Fase 5 - Validacao de Artefato: Blocked (forneca --firmware=<arquivo.bin>)
- Fase 6 - Flash Seguro (Preflight): Blocked (requer modo live)
- Fase 7 - Diagnostico e Debug: Passed (uds_retry=3 p2=1000 p2*= 5000)
- Fase 8 - Governanca e Seguranca: Failed (allowlist_present=false noninteractive_approved=false)
- Fase 9 - Confiabilidade e Stress: Passed (rate_global=100 rate_per_id=10 reconnect=500..30000 ms)
- Fase 10 - Industrializacao: Passed (pipeline de build/test encontrado)
- Fase 11 - Homologacao Final: Passed (dossie json/markdown sera gerado)
- Fase 12 - Gate de Producao: Failed (ha falhas bloqueantes nas fases anteriores)
```

# File Deep Dive: reports/production_phase_report_1786621988419.json

Role: project support artifact

Line count: 87 | Non-empty: 87 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Content Snapshot


```
{
  "generated_at_ms": 1786621988419,
  "mode": "sim",
  "target_sa": null,
  "strict_mode": true,
  "execute_flash": false,
  "artifact": null,
  "detect": null,
  "preflight": null,
  "identity": null,
  "phases": [
    {
      "phase": 1,
      "title": "Baseline e Compliance",
      "status": "Passed",
      "details": "runbook e protocolo presentes"
    },
    {
      "phase": 2,
      "title": "Plataforma Fisica Real",
      "status": "Passed",
      "details": "sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh"
    },
    {
      "phase": 3,
      "title": "Conexao Confiavel",
      "status": "Passed",
      "details": "can_health=0.924 errors=0"
    },
    {
      "phase": 4,
      "title": "Leitura e Identidade ECM",
      "status": "Passed",
      "details": "vin_ok=true sw_ok=true vin_len=20 sw_len=15"
    },
    {
      "phase": 5,
      "title": "Validacao de Artefato",
      "status": "Failed",
```

```
      "details": "forneca --firmware=<arquivo.bin>"
    },
    {
      "phase": 6,
      "title": "Flash Seguro (Preflight)",
      "status": "Passed",
      "details": "battery_v=12.94"
    },
    {
      "phase": 7,
      "title": "Diagnostico e Debug",
      "status": "Failed",
      "details": "uds_retry=3 p2=1000 p2*= 5000 runtime=false: session_sid=0x7F vin_sid=0x62 tester_sid=0x7E"
    },
    {
      "phase": 8,
      "title": "Governanca e Seguranca",
      "status": "Failed",
      "details": "allowlist_present=false noninteractive_approved=false write_effective=false"
    },
    {
      "phase": 9,
      "title": "Confiabilidade e Stress",
      "status": "Passed",
      "details": "rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_speed=0.00 fuel=84.92%"
    },
    {
      "phase": 10,
      "title": "Industrializacao",
      "status": "Passed",
      "details": "pipeline de build/test encontrado"
    },
    {
      "phase": 11,
      "title": "Homologacao Final",
      "status": "Passed",
      "details": "json=reports\\production_phase_report_1786621988419.json md=reports\\production_phase_report_1786621
    },
    {
      "phase": 12,
      "title": "Gate de Producao",
      "status": "Failed",
      "details": "ha falhas bloqueantes nas fases anteriores"
    }
  ],
  "overall_passed": false,
  "report_sha256": "82c565953df96bfe7180abdbc788d978866679335c6c12a7e11d19f69aaa51b0"
}
```


## File Deep Dive: reports/production_phase_report_1786621988419.md

Role: project support artifact

Line count: 21 | Non-empty: 18 | Symbol count: 2

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Production Program Report | namespace and compile-time linkage |
| 8 | h2 | Phase Results | namespace and compile-time linkage |
```

### Content Snapshot

```
# Production Program Report

- generated_at_ms: 1786621988419
- mode: sim
- target_sa: None
- overall_passed: false
```

```
## Phase Results

- Fase 1 - Baseline e Compliance: Passed (runbook e protocolo presentes)
- Fase 2 - Plataforma Fisica Real: Passed (sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh)
- Fase 3 - Conexao Confiavel: Passed (can_health=0.924 errors=0)
- Fase 4 - Leitura e Identidade ECM: Passed (vin_ok=true sw_ok=true vin_len=20 sw_len=15)
- Fase 5 - Validacao de Artefato: Failed (forneca --firmware=<arquivo.bin>)
- Fase 6 - Flash Seguro (Preflight): Passed (battery_v=12.94)
- Fase 7 - Diagnostico e Debug: Failed (uds_retry=3 p2=1000 p2*= 5000 runtime=false: session_sid=0x7F vin_sid=0x62 tes
- Fase 8 - Governanca e Seguranca: Failed (allowlist_present=false noninteractive_approved=false write_effective=false
- Fase 9 - Confiabilidade e Stress: Passed (rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_sp
- Fase 10 - Industrializacao: Passed (pipeline de build/test encontrado)
- Fase 11 - Homologacao Final: Passed (gerando dossie json/markdown)
- Fase 12 - Gate de Producao: Failed (ha falhas bloqueantes nas fases anteriores)
```

# File Deep Dive: reports/production_phase_report_1786622089893.json

Role: project support artifact

Line count: 87 | Non-empty: 87 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Content Snapshot


```
{
  "generated_at_ms": 1786622089893,
  "mode": "sim",
  "target_sa": null,
  "strict_mode": true,
  "execute_flash": false,
  "artifact": null,
  "detect": null,
  "preflight": null,
  "identity": null,
  "phases": [
    {
      "phase": 1,
      "title": "Baseline e Compliance",
      "status": "Passed",
      "details": "runbook e protocolo presentes"
    },
    {
      "phase": 2,
      "title": "Plataforma Fisica Real",
      "status": "Passed",
      "details": "sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh"
    },
    {
      "phase": 3,
      "title": "Conexao Confiavel",
      "status": "Passed",
      "details": "can_health=0.924 errors=0"
    },
    {
      "phase": 4,
      "title": "Leitura e Identidade ECM",
      "status": "Passed",
      "details": "vin_ok=true sw_ok=true vin_len=20 sw_len=15"
    },
    {
      "phase": 5,
      "title": "Validacao de Artefato",
      "status": "Failed",
      "details": "forneca --firmware=<arquivo.bin>"
    },
```

```
    {
      "phase": 6,
      "title": "Flash Seguro (Preflight)",
      "status": "Passed",
      "details": "battery_v=12.94"
    },
    {
      "phase": 7,
      "title": "Diagnostico e Debug",
      "status": "Failed",
      "details": "uds_retry=3 p2=1000 p2*= 5000 runtime=false: session_sid=0x7F vin_sid=0x62 tester_sid=0x7E"
    },
    {
      "phase": 8,
      "title": "Governanca e Seguranca",
      "status": "Failed",
      "details": "allowlist_present=false noninteractive_approved=false write_effective=false"
    },
    {
      "phase": 9,
      "title": "Confiabilidade e Stress",
      "status": "Passed",
      "details": "rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_speed=0.00 fuel=84.92%"
    },
    {
      "phase": 10,
      "title": "Industrializacao",
      "status": "Passed",
      "details": "pipeline de build/test encontrado"
    },
    {
      "phase": 11,
      "title": "Homologacao Final",
      "status": "Passed",
      "details": "json=reports\\production_phase_report_1786622089893.json md=reports\\production_phase_report_1786622
    },
    {
      "phase": 12,
      "title": "Gate de Producao",
      "status": "Failed",
      "details": "ha falhas bloqueantes nas fases anteriores"
    }
  ],
  "overall_passed": false,
  "report_sha256": "420e9d3c32bdcc7b796ac83489f2f3229f8aeb29b3473489b062d8cc394d35db"
}
```


# File Deep Dive: reports/production_phase_report_1786622089893.md

Role: project support artifact

Line count: 21 | Non-empty: 18 | Symbol count: 2

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Production Program Report | namespace and compile-time linkage |
| 8 | h2 | Phase Results | namespace and compile-time linkage |
```

## Content Snapshot

```
# Production Program Report

- generated_at_ms: 1786622089893
- mode: sim
- target_sa: None
- overall_passed: false

## Phase Results
```

```
- Fase 1 - Baseline e Compliance: Passed (runbook e protocolo presentes)
- Fase 2 - Plataforma Fisica Real: Passed (sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh)
- Fase 3 - Conexao Confiavel: Passed (can_health=0.924 errors=0)
- Fase 4 - Leitura e Identidade ECM: Passed (vin_ok=true sw_ok=true vin_len=20 sw_len=15)
- Fase 5 - Validacao de Artefato: Failed (forneca --firmware=<arquivo.bin>)
- Fase 6 - Flash Seguro (Preflight): Passed (battery_v=12.94)
- Fase 7 - Diagnostico e Debug: Failed (uds_retry=3 p2=1000 p2*= 5000 runtime=false: session_sid=0x7F vin_sid=0x62 tes
- Fase 8 - Governanca e Seguranca: Failed (allowlist_present=false noninteractive_approved=false write_effective=false
- Fase 9 - Confiabilidade e Stress: Passed (rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_sp
- Fase 10 - Industrializacao: Passed (pipeline de build/test encontrado)
- Fase 11 - Homologacao Final: Passed (gerando dossie json/markdown)
- Fase 12 - Gate de Producao: Failed (ha falhas bloqueantes nas fases anteriores)
```

## File Deep Dive: reports/production_phase_report_1786622112533.json

Role: project support artifact

Line count: 93 | Non-empty: 93 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Content Snapshot


```
{
  "generated_at_ms": 1786622112533,
  "mode": "sim",
  "target_sa": null,
  "strict_mode": true,
  "execute_flash": false,
  "artifact": {
    "path": "reports/firmware_sample.bin",
    "bytes": 64,
    "crc32": 679541657,
    "sha256": "20a7ec84684f7fe124cb3727d049734ab0b7da2f52fcafbcef989ecfd91e870b",
    "valid": true
  },
  "detect": null,
  "preflight": null,
  "identity": null,
  "phases": [
    {
      "phase": 1,
      "title": "Baseline e Compliance",
      "status": "Passed",
      "details": "runbook e protocolo presentes"
    },
    {
      "phase": 2,
      "title": "Plataforma Fisica Real",
      "status": "Passed",
      "details": "sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh"
    },
    {
      "phase": 3,
      "title": "Conexao Confiavel",
      "status": "Passed",
      "details": "can_health=0.924 errors=0"
    },
    {
      "phase": 4,
      "title": "Leitura e Identidade ECM",
      "status": "Passed",
      "details": "vin_ok=true sw_ok=true vin_len=20 sw_len=15"
    },
    {
      "phase": 5,
```

```
      "title": "Validacao de Artefato",
      "status": "Passed",
      "details": "64 bytes crc32=0x2880FB99"
    },
    {
      "phase": 6,
      "title": "Flash Seguro (Preflight)",
      "status": "Passed",
      "details": "battery_v=12.94"
    },
    {
      "phase": 7,
      "title": "Diagnostico e Debug",
      "status": "Failed",
      "details": "uds_retry=3 p2=1000 p2*= 5000 runtime=false: session_sid=0x7F vin_sid=0x62 tester_sid=0x7E"
    },
    {
      "phase": 8,
      "title": "Governanca e Seguranca",
      "status": "Passed",
      "details": "allowlist_present=true noninteractive_approved=true write_effective=false"
    },
    {
      "phase": 9,
      "title": "Confiabilidade e Stress",
      "status": "Passed",
      "details": "rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_speed=0.00 fuel=84.92%"
    },
    {
      "phase": 10,
      "title": "Industrializacao",
      "status": "Passed",
      "details": "pipeline de build/test encontrado"
    },
    {
      "phase": 11,
      "title": "Homologacao Final",
      "status": "Passed",
      "details": "json=reports\\production_phase_report_1786622112533.json md=reports\\production_phase_report_1786622
    },
    {
      "phase": 12,
      "title": "Gate de Producao",
      "status": "Failed",
      "details": "ha falhas bloqueantes nas fases anteriores"
    }
  ],
  "overall_passed": false,
  "report_sha256": "eb38b2f84c2ed7c5503c4cc7a1ebd62ef983f82cf7934d9989d20e8612db1804"
}
```


## File Deep Dive: reports/production_phase_report_1786622112533.md

Role: project support artifact

Line count: 21 | Non-empty: 18 | Symbol count: 2

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Production Program Report | namespace and compile-time linkage |
| 8 | h2 | Phase Results | namespace and compile-time linkage |
```

### Content Snapshot

```
# Production Program Report

- generated_at_ms: 1786622112533
- mode: sim
```

```
- target_sa: None
- overall_passed: false

## Phase Results

- Fase 1 - Baseline e Compliance: Passed (runbook e protocolo presentes)
- Fase 2 - Plataforma Fisica Real: Passed (sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh)
- Fase 3 - Conexao Confiavel: Passed (can_health=0.924 errors=0)
- Fase 4 - Leitura e Identidade ECM: Passed (vin_ok=true sw_ok=true vin_len=20 sw_len=15)
- Fase 5 - Validacao de Artefato: Passed (64 bytes crc32=0x2880FB99)
- Fase 6 - Flash Seguro (Preflight): Passed (battery_v=12.94)
- Fase 7 - Diagnostico e Debug: Failed (uds_retry=3 p2=1000 p2*= 5000 runtime=false: session_sid=0x7F vin_sid=0x62 tes
- Fase 8 - Governanca e Seguranca: Passed (allowlist_present=true noninteractive_approved=true write_effective=false)
- Fase 9 - Confiabilidade e Stress: Passed (rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_sp
- Fase 10 - Industrializacao: Passed (pipeline de build/test encontrado)
- Fase 11 - Homologacao Final: Passed (gerando dossie json/markdown)
- Fase 12 - Gate de Producao: Failed (ha falhas bloqueantes nas fases anteriores)
```

## File Deep Dive: reports/production_phase_report_1786622154581.json

Role: project support artifact

Line count: 93 | Non-empty: 93 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Content Snapshot


```
{
  "generated_at_ms": 1786622154581,
  "mode": "sim",
  "target_sa": null,
  "strict_mode": true,
  "execute_flash": false,
  "artifact": {
    "path": "reports/firmware_sample.bin",
    "bytes": 64,
    "crc32": 679541657,
    "sha256": "20a7ec84684f7fe124cb3727d049734ab0b7da2f52fcafbcef989ecfd91e870b",
    "valid": true
  },
  "detect": null,
  "preflight": null,
  "identity": null,
  "phases": [
    {
      "phase": 1,
      "title": "Baseline e Compliance",
      "status": "Passed",
      "details": "runbook e protocolo presentes"
    },
    {
      "phase": 2,
      "title": "Plataforma Fisica Real",
      "status": "Passed",
      "details": "sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh"
    },
    {
      "phase": 3,
      "title": "Conexao Confiavel",
      "status": "Passed",
      "details": "can_health=0.924 errors=0"
    },
    {
      "phase": 4,
      "title": "Leitura e Identidade ECM",
```

```
      "status": "Passed",
      "details": "vin_ok=true sw_ok=true vin_len=20 sw_len=15"
    },
    {
      "phase": 5,
      "title": "Validacao de Artefato",
      "status": "Passed",
      "details": "64 bytes crc32=0x2880FB99"
    },
    {
      "phase": 6,
      "title": "Flash Seguro (Preflight)",
      "status": "Passed",
      "details": "battery_v=12.94"
    },
    {
      "phase": 7,
      "title": "Diagnostico e Debug",
      "status": "Passed",
      "details": "uds_retry=3 p2=1000 p2*= 5000 runtime=true: session_sid=0x50 vin_sid=0x62 tester_sid=0x7E"
    },
    {
      "phase": 8,
      "title": "Governanca e Seguranca",
      "status": "Passed",
      "details": "allowlist_present=true noninteractive_approved=true write_effective=false"
    },
    {
      "phase": 9,
      "title": "Confiabilidade e Stress",
      "status": "Passed",
      "details": "rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_speed=0.00 fuel=84.92%"
    },
    {
      "phase": 10,
      "title": "Industrializacao",
      "status": "Passed",
      "details": "pipeline de build/test encontrado"
    },
    {
      "phase": 11,
      "title": "Homologacao Final",
      "status": "Passed",
      "details": "json=reports\\production_phase_report_1786622154581.json md=reports\\production_phase_report_1786622
    },
    {
      "phase": 12,
      "title": "Gate de Producao",
      "status": "Passed",
      "details": "gates criticos satisfeitos para o contexto atual"
    }
  ],
  "overall_passed": true,
  "report_sha256": "b1af20e6479157e3c640cc58fcf72cf469ccf046348fe1275b38fcce28a135ce"
}
```


# File Deep Dive: reports/production_phase_report_1786622154581.md

Role: project support artifact

Line count: 21 | Non-empty: 18 | Symbol count: 2

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Production Program Report | namespace and compile-time linkage |
| 8 | h2 | Phase Results | namespace and compile-time linkage |
```

### Content Snapshot

```
# Production Program Report

- generated_at_ms: 1786622154581
- mode: sim
- target_sa: None
- overall_passed: true

## Phase Results

- Fase 1 - Baseline e Compliance: Passed (runbook e protocolo presentes)
- Fase 2 - Plataforma Fisica Real: Passed (sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh)
- Fase 3 - Conexao Confiavel: Passed (can_health=0.924 errors=0)
- Fase 4 - Leitura e Identidade ECM: Passed (vin_ok=true sw_ok=true vin_len=20 sw_len=15)
- Fase 5 - Validacao de Artefato: Passed (64 bytes crc32=0x2880FB99)
- Fase 6 - Flash Seguro (Preflight): Passed (battery_v=12.94)
- Fase 7 - Diagnostico e Debug: Passed (uds_retry=3 p2=1000 p2*= 5000 runtime=true: session_sid=0x50 vin_sid=0x62 test
- Fase 8 - Governanca e Seguranca: Passed (allowlist_present=true noninteractive_approved=true write_effective=false)
- Fase 9 - Confiabilidade e Stress: Passed (rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_sp
- Fase 10 - Industrializacao: Passed (pipeline de build/test encontrado)
- Fase 11 - Homologacao Final: Passed (gerando dossie json/markdown)
- Fase 12 - Gate de Producao: Passed (gates criticos satisfeitos para o contexto atual)
```

## File Deep Dive: reports/production_phase_report_1786622968863.json

Role: project support artifact

Line count: 112 | Non-empty: 112 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Content Snapshot


```
{
  "generated_at_ms": 1786622968863,
  "mode": "sim",
  "target_sa": null,
  "strict_mode": false,
  "execute_flash": false,
  "artifact": null,
  "detect": null,
  "preflight": null,
  "identity": null,
  "conformance_summary": {
    "passed": true,
    "services": [
      {
        "service_sid": "0x27",
        "passed": true,
        "evidence": "flash path not executed"
      },
      {
        "service_sid": "0x34",
        "passed": true,
        "evidence": "flash path not executed"
      },
      {
        "service_sid": "0x36",
        "passed": true,
        "evidence": "flash path not executed"
      },
      {
        "service_sid": "0x37",
        "passed": true,
        "evidence": "flash path not executed"
      }
```

```
    ]
  },
  "phases": [
    {
      "phase": 1,
      "title": "Baseline e Compliance",
      "status": "Passed",
      "details": "runbook e protocolo presentes"
    },
    {
      "phase": 2,
      "title": "Plataforma Fisica Real",
      "status": "Passed",
      "details": "sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh"
    },
    {
      "phase": 3,
      "title": "Conexao Confiavel",
      "status": "Passed",
      "details": "can_health=0.924 errors=0"
    },
    {
      "phase": 4,
      "title": "Leitura e Identidade ECM",
      "status": "Passed",
      "details": "vin_ok=true sw_ok=true vin_len=20 sw_len=15"
    },
    {
      "phase": 5,
      "title": "Validacao de Artefato",
      "status": "Blocked",
      "details": "forneca --firmware=<arquivo.bin>"
    },
    {
      "phase": 6,
      "title": "Flash Seguro (Preflight)",
      "status": "Passed",
      "details": "battery_v=12.94"
    },
    {
      "phase": 7,
      "title": "Diagnostico e Debug",
      "status": "Passed",
      "details": "uds_retry=3 p2=1000 p2*= 5000 runtime=true: session_sid=0x50 vin_sid=0x62 tester_sid=0x7E"
    },
    {
      "phase": 8,
      "title": "Governanca e Seguranca",
      "status": "Failed",
      "details": "allowlist_present=false noninteractive_approved=false write_effective=false"
    },
    {
      "phase": 9,
      "title": "Confiabilidade e Stress",
      "status": "Passed",
      "details": "rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_speed=0.00 fuel=84.92%"
    },
    {
      "phase": 10,
      "title": "Industrializacao",
      "status": "Passed",
      "details": "pipeline de build/test encontrado"
    },
    {
      "phase": 11,
      "title": "Homologacao Final",
      "status": "Passed",
      "details": "json=reports\\production_phase_report_1786622968863.json md=reports\\production_phase_report_178662
    },
    {
      "phase": 12,
```

```
      "title": "Gate de Producao",
      "status": "Failed",
      "details": "ha falhas bloqueantes nas fases anteriores"
    }
  ],
  "overall_passed": false,
  "report_sha256": "bf8ae41987e0e9d459685177db391d7648078499dda7dcb3e206bd971ba0ebc0"
}
```


# File Deep Dive: reports/production_phase_report_1786622968863.md

Role: project support artifact

Line count: 29 | Non-empty: 24 | Symbol count: 3

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Production Program Report | namespace and compile-time linkage |
| 8 | h2 | UDS Conformance Summary | namespace and compile-time linkage |
| 16 | h2 | Phase Results | namespace and compile-time linkage |
```

## Content Snapshot

```
# Production Program Report

- generated_at_ms: 1786622968863
- mode: sim
- target_sa: None
- overall_passed: false

## UDS Conformance Summary

- passed: true
- SID 0x27: passed=true (flash path not executed)
- SID 0x34: passed=true (flash path not executed)
- SID 0x36: passed=true (flash path not executed)
- SID 0x37: passed=true (flash path not executed)

## Phase Results

- Fase 1 - Baseline e Compliance: Passed (runbook e protocolo presentes)
- Fase 2 - Plataforma Fisica Real: Passed (sim_harness=true: elapsed=6.00s steps=360 speed=0.00kmh)
- Fase 3 - Conexao Confiavel: Passed (can_health=0.924 errors=0)
- Fase 4 - Leitura e Identidade ECM: Passed (vin_ok=true sw_ok=true vin_len=20 sw_len=15)
- Fase 5 - Validacao de Artefato: Blocked (forneca --firmware=<arquivo.bin>)
- Fase 6 - Flash Seguro (Preflight): Passed (battery_v=12.94)
- Fase 7 - Diagnostico e Debug: Passed (uds_retry=3 p2=1000 p2*= 5000 runtime=true: session_sid=0x50 vin_sid=0x62 test
- Fase 8 - Governanca e Seguranca: Failed (allowlist_present=false noninteractive_approved=false write_effective=false
- Fase 9 - Confiabilidade e Stress: Passed (rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_sp
- Fase 10 - Industrializacao: Passed (pipeline de build/test encontrado)
- Fase 11 - Homologacao Final: Passed (gerando dossie json/markdown)
- Fase 12 - Gate de Producao: Failed (ha falhas bloqueantes nas fases anteriores)
```

# File Deep Dive: reports/production_phase_report_1786623531098.json

Role: project support artifact

Line count: 140 | Non-empty: 140 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Content Snapshot

```
{
  "generated_at_ms": 1786623531098,
  "mode": "live",
  "target_sa": 0,
  "strict_mode": true,
  "execute_flash": true,
  "artifact": {
    "path": "C:\\cat\\template\\firmware.bin",
    "bytes": 1024,
    "crc32": 2078597072,
    "sha256": "2bce1ba628720664be4b9fdd77aae0678e5f0f3f02fc6ff641ec879094f6a404",
    "valid": true
  },
  "detect": {
    "source_addresses": [
      0
    ]
  },
  "preflight": {
    "supply_voltage_v": 12.5,
    "identity": {
      "vin": "CATVIN1234567890",
      "sw_version": "SW-1.0.0",
      "hw_version": "HW-1.0.0",
      "calibration_id": "CAL-001",
      "ecu_serial": "ECU-SN-0001",
      "fingerprint": "FP-TRUSTED-01"
    },
    "trusted_fingerprint": true
  },
  "identity": {
    "vin": "CATVIN1234567890",
    "sw_version": "SW-1.0.0",
    "hw_version": "HW-1.0.0",
    "calibration_id": "CAL-001",
    "ecu_serial": "ECU-SN-0001",
    "fingerprint": "FP-TRUSTED-01"
  },
  "conformance_summary": {
    "passed": true,
    "services": [
      {
        "service_sid": "0x27",
        "passed": true,
        "evidence": "seed_ok=true unlock_ok=true"
      },
      {
        "service_sid": "0x34",
        "passed": true,
        "evidence": "request_download_positive=true"
      },
      {
        "service_sid": "0x36",
        "passed": true,
        "evidence": "acked=17/17 seq_err=0 fc_timeout=0 stmin_ms=Some(5) bs=Some(8)"
      },
      {
        "service_sid": "0x37",
        "passed": true,
        "evidence": "request_transfer_exit_positive=true"
      }
    ]
  },
  "phases": [
    {
      "phase": 1,
      "title": "Baseline e Compliance",
      "status": "Passed",
      "details": "runbook e protocolo presentes"
    },
    {
```

```
      "phase": 2,
      "title": "Plataforma Fisica Real",
      "status": "Passed",
      "details": "capabilities: can_raw=false isotp=true j1939=false uds_flash=true vendor_bridge=true"
    },
    {
      "phase": 3,
      "title": "Conexao Confiavel",
      "status": "Passed",
      "details": "ECMs detectados: 1"
    },
    {
      "phase": 4,
      "title": "Leitura e Identidade ECM",
      "status": "Passed",
      "details": "fingerprint=true allow_untrusted_ecu=false"
    },
    {
      "phase": 5,
      "title": "Validacao de Artefato",
      "status": "Passed",
      "details": "1024 bytes crc32=0x7BE4DFD0"
    },
    {
      "phase": 6,
      "title": "Flash Seguro (Preflight)",
      "status": "Passed",
      "details": "supply_v=12.50 trusted_fp=true | flash_ok bytes=1024 blocks=17 crc32=0x7BE4DFD0 | td_ack=17/17 seq_e
    },
    {
      "phase": 7,
      "title": "Diagnostico e Debug",
      "status": "Passed",
      "details": "uds_retry=3 p2=1000 p2*= 5000 runtime=true: session_sid=0x50 vin_sid=0x62 tester_sid=0x7E"
    },
    {
      "phase": 8,
      "title": "Governanca e Seguranca",
      "status": "Passed",
      "details": "allowlist_present=true noninteractive_approved=true write_effective=true"
    },
    {
      "phase": 9,
      "title": "Confiabilidade e Stress",
      "status": "Passed",
      "details": "rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_speed=0.00 fuel=84.92%"
    },
    {
      "phase": 10,
      "title": "Industrializacao",
      "status": "Passed",
      "details": "pipeline de build/test encontrado"
    },
    {
      "phase": 11,
      "title": "Homologacao Final",
      "status": "Passed",
      "details": "json=reports\\production_phase_report_1786623531098.json md=reports\\production_phase_report_178662
    },
    {
      "phase": 12,
      "title": "Gate de Producao",
      "status": "Passed",
      "details": "gates criticos satisfeitos para o contexto atual"
    }
  ],
  "overall_passed": true,
  "report_sha256": "7d9d54c273b7464a27739228b9473bfb2f99ba9e83540b9cd1cef4073abb88db"
}
```

# File Deep Dive: reports/production_phase_report_1786623531098.md

Role: project support artifact

Line count: 29 | Non-empty: 24 | Symbol count: 3

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | h1 | Production Program Report | namespace and compile-time linkage |
| 8 | h2 | UDS Conformance Summary | namespace and compile-time linkage |
| 16 | h2 | Phase Results | namespace and compile-time linkage |
```

## Content Snapshot

```
# Production Program Report

- generated_at_ms: 1786623531098
- mode: live
- target_sa: Some(0)
- overall_passed: true

## UDS Conformance Summary

- passed: true
- SID 0x27: passed=true (seed_ok=true unlock_ok=true)
- SID 0x34: passed=true (request_download_positive=true)
- SID 0x36: passed=true (acked=17/17 seq_err=0 fc_timeout=0 stmin_ms=Some(5) bs=Some(8))
- SID 0x37: passed=true (request_transfer_exit_positive=true)

## Phase Results

- Fase 1 - Baseline e Compliance: Passed (runbook e protocolo presentes)
- Fase 2 - Plataforma Fisica Real: Passed (capabilities: can_raw=false isotp=true j1939=false uds_flash=true vendor_br
- Fase 3 - Conexao Confiavel: Passed (ECMs detectados: 1)
- Fase 4 - Leitura e Identidade ECM: Passed (fingerprint=true allow_untrusted_ecu=false)
- Fase 5 - Validacao de Artefato: Passed (1024 bytes crc32=0x7BE4DFD0)
- Fase 6 - Flash Seguro (Preflight): Passed (supply_v=12.50 trusted_fp=true | flash_ok bytes=1024 blocks=17 crc32=0x7B
- Fase 7 - Diagnostico e Debug: Passed (uds_retry=3 p2=1000 p2*= 5000 runtime=true: session_sid=0x50 vin_sid=0x62 test
- Fase 8 - Governanca e Seguranca: Passed (allowlist_present=true noninteractive_approved=true write_effective=true)
- Fase 9 - Confiabilidade e Stress: Passed (rate_global=100 rate_per_id=10 reconnect=500..30000 ms stress=true: max_sp
- Fase 10 - Industrializacao: Passed (pipeline de build/test encontrado)
- Fase 11 - Homologacao Final: Passed (gerando dossie json/markdown)
- Fase 12 - Gate de Producao: Passed (gates criticos satisfeitos para o contexto atual)
```

# File Deep Dive: scripts/build_cat_comm_template_bridge.ps1

Role: automation and reproducible operational workflows

Line count: 30 | Non-empty: 23 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | param( | support logic; impact depends on neighboring block and data flow. |
| 2 |     [string]$TemplateDir = "C:/cat/template", | support logic; impact depends on neighboring block and data flow.
| 3 |     [switch]$Release | support logic; impact depends on neighboring block and data flow. |
| 4 | ) | support logic; impact depends on neighboring block and data flow. |
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | $ErrorActionPreference = "Stop" | support logic; impact depends on neighboring block and data flow. |
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | $mode = if ($Release) { "--release" } else { "" } | support logic; impact depends on neighboring block and data fl
```

```
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | Write-Host "[1/3] Building cat_comm_bridge (vendor-windows feature)" | support logic; impact depends on neighbori
| 11 | if ($mode -eq "--release") { | runtime gate; condition changes can enable/disable critical paths. |
| 12 |     cargo build --features vendor-windows --bin cat_comm_bridge --release | support logic; impact depends on neig
| 13 |     $src = Join-Path (Get-Location) "target/release/cat_comm_bridge.exe" | support logic; impact depends on neigh
| 14 | } else { | support logic; impact depends on neighboring block and data flow. |
| 15 |     cargo build --features vendor-windows --bin cat_comm_bridge | support logic; impact depends on neighboring b
| 16 |     $src = Join-Path (Get-Location) "target/debug/cat_comm_bridge.exe" | support logic; impact depends on neighbo
| 17 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 18 | (blank) | blank separator; no runtime behavior. |
| 19 | if (-not (Test-Path $src)) { | runtime gate; condition changes can enable/disable critical paths. |
| 20 |     throw "Bridge executable not found at $src" | support logic; impact depends on neighboring block and data flo
| 21 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 22 | (blank) | blank separator; no runtime behavior. |
| 23 | Write-Host "[2/3] Preparing template dir $TemplateDir" | support logic; impact depends on neighboring block and d
| 24 | New-Item -ItemType Directory -Path $TemplateDir -Force \| Out-Null | support logic; impact depends on neighboring
| 25 | $dst = Join-Path $TemplateDir "cat_comm_bridge.exe" | support logic; impact depends on neighboring block and data
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 | Write-Host "[3/3] Copying executable" | support logic; impact depends on neighboring block and data flow. |
| 28 | Copy-Item -Path $src -Destination $dst -Force | support logic; impact depends on neighboring block and data flow
| 29 | (blank) | blank separator; no runtime behavior. |
| 30 | Write-Host "DONE: $dst" | support logic; impact depends on neighboring block and data flow. |
```

# File Deep Dive: scripts/replay_from_artifact.sh

Role: automation and reproducible operational workflows

Line count: 11 | Non-empty: 9 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #!/usr/bin/env bash | comment/documentation; changing affects understanding and intent traceability. |
| 2 | set -euo pipefail | support logic; impact depends on neighboring block and data flow. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | artifact=${1:-} | support logic; impact depends on neighboring block and data flow. |
| 5 | if [[ -z "$artifact" ]]; then | runtime gate; condition changes can enable/disable critical paths. |
| 6 |   echo "usage: $0 <artifact_log>" | support logic; impact depends on neighboring block and data flow. |
| 7 |   exit 1 | support logic; impact depends on neighboring block and data flow. |
| 8 | fi | support logic; impact depends on neighboring block and data flow. |
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | cargo build --release --bin simulator_cli | support logic; impact depends on neighboring block and data flow. |
| 11 | ./target/release/simulator_cli --replay "$artifact" --verify | support logic; impact depends on neighboring bloc
```

# File Deep Dive: scripts/run_fuzz_smoke.sh

Role: automation and reproducible operational workflows

Line count: 9 | Non-empty: 7 | Symbol count: 0

## Symbol Index

No symbols extracted for this file type.

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #!/usr/bin/env bash | comment/documentation; changing affects understanding and intent traceability. |
| 2 | set -euo pipefail | support logic; impact depends on neighboring block and data flow. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | if ! command -v cargo-fuzz >/dev/null 2>&1; then | runtime gate; condition changes can enable/disable critical pa
| 5 |   cargo install cargo-fuzz | support logic; impact depends on neighboring block and data flow. |
| 6 | fi | support logic; impact depends on neighboring block and data flow. |
```

```
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | # Keep CI runtime bounded. | comment/documentation; changing affects understanding and intent traceability. |
| 9 | cargo fuzz run j1939_frame_from_raw -- -max_total_time=120 | support logic; impact depends on neighboring block an
```

## File Deep Dive: scripts/run_integration.sh

Role: automation and reproducible operational workflows

Line count: 4 | Non-empty: 3 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #!/usr/bin/env bash | comment/documentation; changing affects understanding and intent traceability. |
| 2 | set -euo pipefail | support logic; impact depends on neighboring block and data flow. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | cargo test --test leak_system_integration | support logic; impact depends on neighboring block and data flow. |
```

## File Deep Dive: scripts/run_monte_carlo.sh

Role: automation and reproducible operational workflows

Line count: 17 | Non-empty: 13 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #!/usr/bin/env bash | comment/documentation; changing affects understanding and intent traceability. |
| 2 | set -euo pipefail | support logic; impact depends on neighboring block and data flow. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | seeds=${1:-5} | support logic; impact depends on neighboring block and data flow. |
| 5 | model=${2:-reduced} | support logic; impact depends on neighboring block and data flow. |
| 6 | out_dir=${3:-artifacts/monte} | support logic; impact depends on neighboring block and data flow. |
| 7 | mkdir -p "$out_dir" | support logic; impact depends on neighboring block and data flow. |
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | cargo build --release --bin simulator_cli | support logic; impact depends on neighboring block and data flow. |
| 10 | (blank) | blank separator; no runtime behavior. |
| 11 | for i in $(seq 1 "$seeds"); do | iteration logic; may alter timing, throughput, and safety constraints. |
| 12 |   seed=$(( (RANDOM<<16) ^ i )) | support logic; impact depends on neighboring block and data flow. |
| 13 |   log_file="$out_dir/monte_${model}_${seed}.log" | support logic; impact depends on neighboring block and data fl
| 14 |   ./target/release/simulator_cli --seed "$seed" --model "$model" --steps 10000 2>&1 \| tee "$log_file" \|\| true
| 15 | done | support logic; impact depends on neighboring block and data flow. |
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 | echo "Monte Carlo run complete: $out_dir" | support logic; impact depends on neighboring block and data flow. |
```

## File Deep Dive: scripts/run_windows_template_e2e.ps1

Role: automation and reproducible operational workflows

Line count: 42 | Non-empty: 34 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | param( | support logic; impact depends on neighboring block and data flow. |
| 2 |     [string]$TemplateDir = "C:/cat/template", | support logic; impact depends on neighboring block and data flow.
| 3 |     [string]$FirmwarePath = "", | support logic; impact depends on neighboring block and data flow. |
| 4 |     [string]$ReportDir = "reports" | support logic; impact depends on neighboring block and data flow. |
| 5 | ) | support logic; impact depends on neighboring block and data flow. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | $ErrorActionPreference = "Stop" | support logic; impact depends on neighboring block and data flow. |
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | if ([string]::IsNullOrWhiteSpace($FirmwarePath)) { | runtime gate; condition changes can enable/disable critical p
| 10 |     $FirmwarePath = Join-Path $TemplateDir "firmware.bin" | support logic; impact depends on neighboring block an
| 11 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | Write-Host "[1/6] Building and staging bridge" | support logic; impact depends on neighboring block and data flow
| 14 | & "$PSScriptRoot/build_cat_comm_template_bridge.ps1" -TemplateDir $TemplateDir | support logic; impact depends on
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | Write-Host "[2/6] Preparing allowlist and firmware" | support logic; impact depends on neighboring block and data
| 17 | New-Item -ItemType Directory -Path $TemplateDir -Force \| Out-Null | support logic; impact depends on neighboring
| 18 | $allowlistPath = Join-Path $TemplateDir "allowlist.json" | support logic; impact depends on neighboring block and
| 19 | if (-not (Test-Path $allowlistPath)) { | runtime gate; condition changes can enable/disable critical paths. |
| 20 |     Set-Content -Path $allowlistPath -Value "[]" | support logic; impact depends on neighboring block and data f
| 21 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 22 | if (-not (Test-Path $FirmwarePath)) { | runtime gate; condition changes can enable/disable critical paths. |
| 23 |     $bytes = New-Object byte[] 1024 | support logic; impact depends on neighboring block and data flow. |
| 24 |     for ($i = 0; $i -lt $bytes.Length; $i++) { $bytes[$i] = [byte]($i % 251) } | iteration logic; may alter timi
| 25 |     [System.IO.File]::WriteAllBytes($FirmwarePath, $bytes) | support logic; impact depends on neighboring block
| 26 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 | (blank) | blank separator; no runtime behavior. |
| 28 | Write-Host "[3/6] Bridge negotiation validation" | support logic; impact depends on neighboring block and data f
| 29 | cargo run --features vendor-windows --bin simulator_cli -- --validate-cat-bridge --hw-mode=live --vendor-name=ca
| 30 | (blank) | blank separator; no runtime behavior. |
| 31 | Write-Host "[4/6] Running full production phases (strict + execute flash)" | protocol or I/O critical; changes m
| 32 | cargo run --features vendor-windows --bin simulator_cli -- --run-production-phases --hw-mode=live --vendor-name=
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 | Write-Host "[5/6] Locating newest report" | support logic; impact depends on neighboring block and data flow. |
| 35 | $report = Get-ChildItem -Path $ReportDir -Filter "production_phase_report_*.json" \| Sort-Object LastWriteTime -
| 36 | if (-not $report) { | runtime gate; condition changes can enable/disable critical paths. |
| 37 |     throw "No production report generated" | support logic; impact depends on neighboring block and data flow. |
| 38 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 | (blank) | blank separator; no runtime behavior. |
| 40 | Write-Host "[6/6] Report summary" | support logic; impact depends on neighboring block and data flow. |
| 41 | Get-Content $report.FullName -Raw \| ConvertFrom-Json \| Select-Object generated_at_ms, mode, overall_passed, re
| 42 | Write-Host "Report file: $($report.FullName)" | support logic; impact depends on neighboring block and data flow
```

## File Deep Dive: scripts/validate_can_utils_interop.sh

Role: automation and reproducible operational workflows

Line count: 63 | Non-empty: 51 | Symbol count: 0

### Symbol Index

No symbols extracted for this file type.

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #!/usr/bin/env bash | comment/documentation; changing affects understanding and intent traceability. |
| 2 | set -euo pipefail | support logic; impact depends on neighboring block and data flow. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | # Linux bench validation script for SocketCAN + ISO-TP interoperability. | comment/documentation; changing affects
| 5 | # Requires can-utils installed and a vcan interface available. | comment/documentation; changing affects understan
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | CAN_IF="${CAN_IF:-vcan0}" | support logic; impact depends on neighboring block and data flow. |
| 8 | SRC_ID="${SRC_ID:-7E0}" | support logic; impact depends on neighboring block and data flow. |
| 9 | DST_ID="${DST_ID:-7E8}" | support logic; impact depends on neighboring block and data flow. |
| 10 | TIMEOUT_SEC="${TIMEOUT_SEC:-5}" | support logic; impact depends on neighboring block and data flow. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | need_cmd() { | support logic; impact depends on neighboring block and data flow. |
```

```
| 13 |   command -v "$1" >/dev/null 2>&1 \|\| { | support logic; impact depends on neighboring block and data flow. |
| 14 |     echo "missing command: $1" >&2 | support logic; impact depends on neighboring block and data flow. |
| 15 |     exit 1 | support logic; impact depends on neighboring block and data flow. |
| 16 |   } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 17 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 18 | (blank) | blank separator; no runtime behavior. |
| 19 | need_cmd ip | support logic; impact depends on neighboring block and data flow. |
| 20 | need_cmd candump | support logic; impact depends on neighboring block and data flow. |
| 21 | need_cmd isotpsend | protocol or I/O critical; changes may impact external integration correctness. |
| 22 | need_cmd isotprecv | support logic; impact depends on neighboring block and data flow. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | echo "[1/5] Checking CAN interface ${CAN_IF}" | support logic; impact depends on neighboring block and data flow
| 25 | if ! ip link show "${CAN_IF}" >/dev/null 2>&1; then | runtime gate; condition changes can enable/disable critical
| 26 |   echo "interface ${CAN_IF} not found" >&2 | support logic; impact depends on neighboring block and data flow. |
| 27 |   exit 1 | support logic; impact depends on neighboring block and data flow. |
| 28 | fi | support logic; impact depends on neighboring block and data flow. |
| 29 | (blank) | blank separator; no runtime behavior. |
| 30 | echo "[2/5] Starting capture window" | support logic; impact depends on neighboring block and data flow. |
| 31 | TMP_LOG="$(mktemp /tmp/autobreaking_canutils_XXXXXX.log)" | support logic; impact depends on neighboring block an
| 32 | (candump "${CAN_IF}" >"${TMP_LOG}" 2>&1 &) | support logic; impact depends on neighboring block and data flow. |
| 33 | CANDUMP_PID=$! | support logic; impact depends on neighboring block and data flow. |
| 34 | trap 'kill ${CANDUMP_PID} >/dev/null 2>&1 \|\| true; rm -f "${TMP_LOG}"' EXIT | support logic; impact depends on
| 35 | sleep 0.2 | support logic; impact depends on neighboring block and data flow. |
| 36 | (blank) | blank separator; no runtime behavior. |
| 37 | echo "[3/5] Launching ISO-TP receiver" | support logic; impact depends on neighboring block and data flow. |
| 38 | TMP_RX="$(mktemp /tmp/autobreaking_isotp_rx_XXXXXX.log)" | support logic; impact depends on neighboring block and
| 39 | (timeout "${TIMEOUT_SEC}" isotprecv -s "${DST_ID}" -d "${SRC_ID}" "${CAN_IF}" >"${TMP_RX}" 2>&1 &) | support logi
| 40 | RX_PID=$! | support logic; impact depends on neighboring block and data flow. |
| 41 | sleep 0.2 | support logic; impact depends on neighboring block and data flow. |
| 42 | (blank) | blank separator; no runtime behavior. |
| 43 | echo "[4/5] Sending UDS TesterPresent over ISO-TP" | protocol or I/O critical; changes may impact external integ
| 44 | printf '3E 00\n' \| isotpsend -s "${SRC_ID}" -d "${DST_ID}" "${CAN_IF}" | protocol or I/O critical; changes may
| 45 | (blank) | blank separator; no runtime behavior. |
| 46 | wait "${RX_PID}" \|\| true | support logic; impact depends on neighboring block and data flow. |
| 47 | (blank) | blank separator; no runtime behavior. |
| 48 | kill "${CANDUMP_PID}" >/dev/null 2>&1 \|\| true | support logic; impact depends on neighboring block and data fl
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 | if grep -Eq '3E 00\|3e 00' "${TMP_RX}"; then | runtime gate; condition changes can enable/disable critical paths
| 51 |   echo "[5/5] PASS: ISO-TP payload observed on receiver" | support logic; impact depends on neighboring block an
| 52 | else | support logic; impact depends on neighboring block and data flow. |
| 53 |   echo "[5/5] FAIL: payload not observed by isotprecv" >&2 | support logic; impact depends on neighboring block
| 54 |   echo "--- isotprecv output ---" >&2 | support logic; impact depends on neighboring block and data flow. |
| 55 |   cat "${TMP_RX}" >&2 \|\| true | support logic; impact depends on neighboring block and data flow. |
| 56 |   echo "--- candump output ---" >&2 | support logic; impact depends on neighboring block and data flow. |
| 57 |   cat "${TMP_LOG}" >&2 \|\| true | support logic; impact depends on neighboring block and data flow. |
| 58 |   exit 1 | support logic; impact depends on neighboring block and data flow. |
| 59 | fi | support logic; impact depends on neighboring block and data flow. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 | echo "Validation logs:" | support logic; impact depends on neighboring block and data flow. |
| 62 | echo "  isotprecv: ${TMP_RX}" | support logic; impact depends on neighboring block and data flow. |
| 63 | echo "  candump:   ${TMP_LOG}" | support logic; impact depends on neighboring block and data flow. |
```

## File Deep Dive: src/adas.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 252 | Non-empty: 219 | Symbol count: 10

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| fmt | 1 |
| new | 1 |
| update | 1 |
| toggle_acc | 1 |
| acc_speed_up | 1 |
| acc_speed_down | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 5 | struct | AdasModule | data/schema coupling and match/serde break risk |
```

```
| 48 | enum | ParkZone | data/schema coupling and match/serde break risk |
| 55 | impl | std::fmt::Display for ParkZone | method behavior and trait semantics |
| 56 | fn | fmt | API and behavior coupling to callers/implementers |
| 66 | impl | AdasModule | method behavior and trait semantics |
| 67 | fn | new | API and behavior coupling to callers/implementers |
| 105 | fn | update | API and behavior coupling to callers/implementers |
| 238 | fn | toggle_acc | API and behavior coupling to callers/implementers |
| 246 | fn | acc_speed_up | API and behavior coupling to callers/implementers |
| 249 | fn | acc_speed_down | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use crate::sensors::SensorSuite; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | /// ADAS domain controller (SAE Level 2) | comment/documentation; changing affects understanding and intent tracea
| 4 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 5 | pub struct AdasModule { | data layout contract; field changes can break callers/serde/tests. |
| 6 |     // Adaptive Cruise Control | comment/documentation; changing affects understanding and intent traceability. |
| 7 |     pub acc_enabled: bool, | support logic; impact depends on neighboring block and data flow. |
| 8 |     pub acc_set_speed: f64,    // km/h | support logic; impact depends on neighboring block and data flow. |
| 9 |     pub acc_headway_time: f64, // s (gap to lead) | support logic; impact depends on neighboring block and data f
| 10 |     pub acc_throttle_out: f64, | support logic; impact depends on neighboring block and data flow. |
| 11 |     pub acc_brake_out: f64, | support logic; impact depends on neighboring block and data flow. |
| 12 |     acc_speed_error_integral: f64, | support logic; impact depends on neighboring block and data flow. |
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 |     // Autonomous Emergency Braking | comment/documentation; changing affects understanding and intent traceabil
| 15 |     pub aeb_enabled: bool, | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub aeb_pre_charge: bool, // brakes pre-charged | support logic; impact depends on neighboring block and data
| 17 |     pub aeb_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 18 |     pub aeb_ttc: f64, // seconds to collision | support logic; impact depends on neighboring block and data flow
| 19 | (blank) | blank separator; no runtime behavior. |
| 20 |     // Lane Keeping Assist / Lane Departure Warning | comment/documentation; changing affects understanding and i
| 21 |     pub lka_enabled: bool, | support logic; impact depends on neighboring block and data flow. |
| 22 |     pub lka_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 23 |     pub lka_steering_correction: f64, | support logic; impact depends on neighboring block and data flow. |
| 24 |     pub ldw_warning: bool, | support logic; impact depends on neighboring block and data flow. |
| 25 | (blank) | blank separator; no runtime behavior. |
| 26 |     // Blind Spot Monitoring | comment/documentation; changing affects understanding and intent traceability. |
| 27 |     pub bsm_enabled: bool, | support logic; impact depends on neighboring block and data flow. |
| 28 |     pub bsm_left_warning: bool, | support logic; impact depends on neighboring block and data flow. |
| 29 |     pub bsm_right_warning: bool, | support logic; impact depends on neighboring block and data flow. |
| 30 | (blank) | blank separator; no runtime behavior. |
| 31 |     // Park Assist (front/rear ultrasonic guidance) | comment/documentation; changing affects understanding and i
| 32 |     pub park_assist_enabled: bool, | support logic; impact depends on neighboring block and data flow. |
| 33 |     pub park_assist_zone: ParkZone, | support logic; impact depends on neighboring block and data flow. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 |     // Traffic Sign Recognition | comment/documentation; changing affects understanding and intent traceability.
| 36 |     pub tsr_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 37 |     pub tsr_speed_limit: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 38 | (blank) | blank separator; no runtime behavior. |
| 39 |     // Driver Attention Monitor (simulated) | comment/documentation; changing affects understanding and intent t
| 40 |     pub dam_drowsiness: f64, // 0-1 | support logic; impact depends on neighboring block and data flow. |
| 41 |     pub dam_alert: bool, | support logic; impact depends on neighboring block and data flow. |
| 42 | (blank) | blank separator; no runtime behavior. |
| 43 |     // Autonomous level: 0=none, 1=partial, 2=combined | comment/documentation; changing affects understanding a
| 44 |     pub autonomy_level: u8, | support logic; impact depends on neighboring block and data flow. |
| 45 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 46 | (blank) | blank separator; no runtime behavior. |
| 47 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 48 | pub enum ParkZone { | state machine variants; changes ripple through matches and business logic. |
| 49 |     Clear, | support logic; impact depends on neighboring block and data flow. |
| 50 |     Caution, | support logic; impact depends on neighboring block and data flow. |
| 51 |     Warning, | support logic; impact depends on neighboring block and data flow. |
| 52 |     Critical, | support logic; impact depends on neighboring block and data flow. |
| 53 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 | (blank) | blank separator; no runtime behavior. |
| 55 | impl std::fmt::Display for ParkZone { | implementation binding; behavior semantics and trait conformance live he
| 56 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 57 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 58 |             ParkZone::Clear => write!(f, "CLEAR"), | protocol or I/O critical; changes may impact external integ
| 59 |             ParkZone::Caution => write!(f, "CAUTION"), | protocol or I/O critical; changes may impact external i
| 60 |             ParkZone::Warning => write!(f, "WARNING"), | protocol or I/O critical; changes may impact external i
| 61 |             ParkZone::Critical => write!(f, "CRITICAL"), | protocol or I/O critical; changes may impact external
| 62 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 63 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 64 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 65 | (blank) | blank separator; no runtime behavior. |
| 66 | impl AdasModule { | implementation binding; behavior semantics and trait conformance live here. |
| 67 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 68 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 69 |             acc_enabled: false, | support logic; impact depends on neighboring block and data flow. |
| 70 |             acc_set_speed: 80.0, | support logic; impact depends on neighboring block and data flow. |
| 71 |             acc_headway_time: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 72 |             acc_throttle_out: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 73 |             acc_brake_out: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 74 |             acc_speed_error_integral: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 75 | (blank) | blank separator; no runtime behavior. |
| 76 |             aeb_enabled: true, | support logic; impact depends on neighboring block and data flow. |
| 77 |             aeb_pre_charge: false, | support logic; impact depends on neighboring block and data flow. |
| 78 |             aeb_active: false, | support logic; impact depends on neighboring block and data flow. |
| 79 |             aeb_ttc: f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 80 | (blank) | blank separator; no runtime behavior. |
| 81 |             lka_enabled: false, | support logic; impact depends on neighboring block and data flow. |
| 82 |             lka_active: false, | support logic; impact depends on neighboring block and data flow. |
| 83 |             lka_steering_correction: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 84 |             ldw_warning: false, | support logic; impact depends on neighboring block and data flow. |
| 85 | (blank) | blank separator; no runtime behavior. |
| 86 |             bsm_enabled: true, | support logic; impact depends on neighboring block and data flow. |
| 87 |             bsm_left_warning: false, | support logic; impact depends on neighboring block and data flow. |
| 88 |             bsm_right_warning: false, | support logic; impact depends on neighboring block and data flow. |
| 89 | (blank) | blank separator; no runtime behavior. |
| 90 |             park_assist_enabled: false, | support logic; impact depends on neighboring block and data flow. |
| 91 |             park_assist_zone: ParkZone::Clear, | support logic; impact depends on neighboring block and data flow
| 92 | (blank) | blank separator; no runtime behavior. |
| 93 |             tsr_active: true, | support logic; impact depends on neighboring block and data flow. |
| 94 |             tsr_speed_limit: Some(80), | support logic; impact depends on neighboring block and data flow. |
| 95 | (blank) | blank separator; no runtime behavior. |
| 96 |             dam_drowsiness: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 97 |             dam_alert: false, | support logic; impact depends on neighboring block and data flow. |
| 98 | (blank) | blank separator; no runtime behavior. |
| 99 |             autonomy_level: 2, | support logic; impact depends on neighboring block and data flow. |
| 100 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 101 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 102 | (blank) | blank separator; no runtime behavior. |
| 103 |     /// Returns (throttle_override, brake_override, steering_override) | comment/documentation; changing affects
| 104 |     /// None means no override for that channel. | comment/documentation; changing affects understanding and int
| 105 |     pub fn update( | function signature/body; changes affect API behavior and control-flow. |
| 106 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 107 |         sensors: &SensorSuite, | support logic; impact depends on neighboring block and data flow. |
| 108 |         vehicle_speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 109 |         _user_throttle: f64, | support logic; impact depends on neighboring block and data flow. |
| 110 |         user_brake: f64, | support logic; impact depends on neighboring block and data flow. |
| 111 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 112 |     ) -> (Option<f64>, Option<f64>, Option<f64>) { | support logic; impact depends on neighboring block and data
| 113 |         let mut throttle_out: Option<f64> = None; | support logic; impact depends on neighboring block and data
| 114 |         let mut brake_out: Option<f64> = None; | support logic; impact depends on neighboring block and data flo
| 115 |         let mut steer_out: Option<f64> = None; | support logic; impact depends on neighboring block and data flo
| 116 | (blank) | blank separator; no runtime behavior. |
| 117 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 118 |         // AEB — highest priority | comment/documentation; changing affects understanding and intent traceabili
| 119 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 120 |         self.aeb_ttc = f64::INFINITY; | support logic; impact depends on neighboring block and data flow. |
| 121 |         self.aeb_pre_charge = false; | support logic; impact depends on neighboring block and data flow. |
| 122 |         self.aeb_active = false; | support logic; impact depends on neighboring block and data flow. |
| 123 | (blank) | blank separator; no runtime behavior. |
| 124 |         if self.aeb_enabled { | runtime gate; condition changes can enable/disable critical paths. |
| 125 |             // Find closest in-path forward target | comment/documentation; changing affects understanding and i
| 126 |             if let Some(target) = sensors | runtime gate; condition changes can enable/disable critical paths.
| 127 |                 .forward_targets | support logic; impact depends on neighboring block and data flow. |
| 128 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 129 |                 .filter(\|t\| t.is_in_path && t.relative_speed > 0.0) | support logic; impact depends on neighbo
| 130 |                 .min_by(\|a, b\| a.distance.total_cmp(&b.distance)) | support logic; impact depends on neighbori
| 131 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 132 |                 let ttc = target.distance / (target.relative_speed / 3.6).max(0.01); | support logic; impact dep
| 133 |                 self.aeb_ttc = ttc; | support logic; impact depends on neighboring block and data flow. |
| 134 | (blank) | blank separator; no runtime behavior. |
| 135 |                 if ttc < 2.5 { | runtime gate; condition changes can enable/disable critical paths. |
| 136 |                     self.aeb_pre_charge = true; | support logic; impact depends on neighboring block and data fl
| 137 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 |                 if ttc < 1.8 { | runtime gate; condition changes can enable/disable critical paths. |
```

```
| 139 |                     // Partial braking | comment/documentation; changing affects understanding and intent tracea
| 140 |                     let partial = ((2.5 - ttc) / 1.5).clamp(0.0, 0.6); | support logic; impact depends on neighb
| 141 |                     brake_out = Some(partial.max(user_brake)); | support logic; impact depends on neighboring b
| 142 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 143 |                 if ttc < 0.9 { | runtime gate; condition changes can enable/disable critical paths. |
| 144 |                     // Full AEB | comment/documentation; changing affects understanding and intent traceability
| 145 |                     self.aeb_active = true; | support logic; impact depends on neighboring block and data flow.
| 146 |                     brake_out = Some(1.0); | support logic; impact depends on neighboring block and data flow.
| 147 |                     throttle_out = Some(0.0); | support logic; impact depends on neighboring block and data flo
| 148 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 149 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 150 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 151 | (blank) | blank separator; no runtime behavior. |
| 152 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 153 |         // ACC — overrides throttle/brake when enabled | comment/documentation; changing affects understanding a
| 154 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 155 |         if self.acc_enabled && !self.aeb_active { | runtime gate; condition changes can enable/disable critical
| 156 |             let lead = sensors | support logic; impact depends on neighboring block and data flow. |
| 157 |                 .forward_targets | support logic; impact depends on neighboring block and data flow. |
| 158 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 159 |                 .filter(\|t\| t.is_in_path) | support logic; impact depends on neighboring block and data flow.
| 160 |                 .min_by(\|a, b\| a.distance.total_cmp(&b.distance)); | support logic; impact depends on neighbo
| 161 | (blank) | blank separator; no runtime behavior. |
| 162 |             let (_target_speed, speed_error) = if let Some(t) = lead { | support logic; impact depends on neigh
| 163 |                 let safe_dist = vehicle_speed / 3.6 * self.acc_headway_time; | support logic; impact depends on
| 164 |                 let dist_error = t.distance - safe_dist; | support logic; impact depends on neighboring block a
| 165 |                 // Blend: follow gap OR speed, whichever is more restrictive | comment/documentation; changing a
| 166 |                 let gap_speed = vehicle_speed + dist_error * 0.4; | support logic; impact depends on neighborin
| 167 |                 let tgt = gap_speed.min(self.acc_set_speed).max(0.0); | support logic; impact depends on neighb
| 168 |                 (tgt, tgt - vehicle_speed) | support logic; impact depends on neighboring block and data flow.
| 169 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 170 |                 (self.acc_set_speed, self.acc_set_speed - vehicle_speed) | support logic; impact depends on nei
| 171 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 172 | (blank) | blank separator; no runtime behavior. |
| 173 |             // PI speed controller | comment/documentation; changing affects understanding and intent traceabil
| 174 |             self.acc_speed_error_integral += speed_error * dt; | support logic; impact depends on neighboring b
| 175 |             self.acc_speed_error_integral = self.acc_speed_error_integral.clamp(-30.0, 30.0); | support logic; i
| 176 |             let control = speed_error * 0.05 + self.acc_speed_error_integral * 0.002; | support logic; impact de
| 177 | (blank) | blank separator; no runtime behavior. |
| 178 |             if control > 0.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 179 |                 self.acc_throttle_out = control.clamp(0.0, 1.0); | support logic; impact depends on neighboring
| 180 |                 self.acc_brake_out = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 181 |                 throttle_out = Some(self.acc_throttle_out); | support logic; impact depends on neighboring bloc
| 182 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 183 |                 self.acc_brake_out = (-control).clamp(0.0, 0.8); | support logic; impact depends on neighboring
| 184 |                 self.acc_throttle_out = 0.0; | support logic; impact depends on neighboring block and data flow
| 185 |                 throttle_out = Some(0.0); | support logic; impact depends on neighboring block and data flow. |
| 186 |                 brake_out = Some(self.acc_brake_out.max(brake_out.unwrap_or(0.0))); | support logic; impact dep
| 187 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 188 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 189 | (blank) | blank separator; no runtime behavior. |
| 190 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 191 |         // LKA — steering correction | comment/documentation; changing affects understanding and intent traceabi
| 192 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 193 |         self.lka_active = false; | support logic; impact depends on neighboring block and data flow. |
| 194 |         self.lka_steering_correction = 0.0; | support logic; impact depends on neighboring block and data flow.
| 195 |         if self.lka_enabled && vehicle_speed > 30.0 { | runtime gate; condition changes can enable/disable crit
| 196 |             let correction = -sensors.lane_offset * 4.0; // proportional | support logic; impact depends on nei
| 197 |             if correction.abs() > 0.3 { | runtime gate; condition changes can enable/disable critical paths. |
| 198 |                 self.lka_active = true; | support logic; impact depends on neighboring block and data flow. |
| 199 |                 self.lka_steering_correction = correction.clamp(-5.0, 5.0); | support logic; impact depends on
| 200 |                 steer_out = Some(self.lka_steering_correction); | support logic; impact depends on neighboring
| 201 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 202 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 203 |         self.ldw_warning = sensors.lane_departure_left \|\| sensors.lane_departure_right; | support logic; impac
| 204 | (blank) | blank separator; no runtime behavior. |
| 205 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 206 |         // BSM | comment/documentation; changing affects understanding and intent traceability. |
| 207 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 208 |         if self.bsm_enabled { | runtime gate; condition changes can enable/disable critical paths. |
| 209 |             self.bsm_left_warning = sensors.bsm_left; | support logic; impact depends on neighboring block and
| 210 |             self.bsm_right_warning = sensors.bsm_right; | support logic; impact depends on neighboring block an
| 211 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 212 | (blank) | blank separator; no runtime behavior. |
| 213 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 214 |         // Park Assist | comment/documentation; changing affects understanding and intent traceability. |
```

```
| 215 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 216 |         let front_min = sensors.ultrasonic[..3] | support logic; impact depends on neighboring block and data f
| 217 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 218 |             .cloned() | support logic; impact depends on neighboring block and data flow. |
| 219 |             .fold(f64::INFINITY, f64::min); | support logic; impact depends on neighboring block and data flow.
| 220 |         self.park_assist_zone = if front_min > 2.0 { | support logic; impact depends on neighboring block and da
| 221 |             ParkZone::Clear | support logic; impact depends on neighboring block and data flow. |
| 222 |         } else if front_min > 1.0 { | support logic; impact depends on neighboring block and data flow. |
| 223 |             ParkZone::Caution | support logic; impact depends on neighboring block and data flow. |
| 224 |         } else if front_min > 0.4 { | support logic; impact depends on neighboring block and data flow. |
| 225 |             ParkZone::Warning | support logic; impact depends on neighboring block and data flow. |
| 226 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 227 |             ParkZone::Critical | support logic; impact depends on neighboring block and data flow. |
| 228 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 229 | (blank) | blank separator; no runtime behavior. |
| 230 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 231 |         // TSR | comment/documentation; changing affects understanding and intent traceability. |
| 232 |         // ---------------------------------------------------------------- | comment/documentation; changing
| 233 |         self.tsr_speed_limit = sensors.speed_limit_kmh; | support logic; impact depends on neighboring block an
| 234 | (blank) | blank separator; no runtime behavior. |
| 235 |         (throttle_out, brake_out, steer_out) | support logic; impact depends on neighboring block and data flow
| 236 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 237 | (blank) | blank separator; no runtime behavior. |
| 238 |     pub fn toggle_acc(&mut self, current_speed: f64) { | function signature/body; changes affect API behavior a
| 239 |         self.acc_enabled = !self.acc_enabled; | support logic; impact depends on neighboring block and data flo
| 240 |         if self.acc_enabled { | runtime gate; condition changes can enable/disable critical paths. |
| 241 |             self.acc_set_speed = current_speed.max(30.0); | support logic; impact depends on neighboring block
| 242 |             self.acc_speed_error_integral = 0.0; | support logic; impact depends on neighboring block and data
| 243 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 244 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 245 | (blank) | blank separator; no runtime behavior. |
| 246 |     pub fn acc_speed_up(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 247 |         self.acc_set_speed = (self.acc_set_speed + 5.0).min(200.0); | support logic; impact depends on neighbor
| 248 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 249 |     pub fn acc_speed_down(&mut self) { | function signature/body; changes affect API behavior and control-flow.
| 250 |         self.acc_set_speed = (self.acc_set_speed - 5.0).max(0.0); | support logic; impact depends on neighborin
| 251 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 252 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/autonomous.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 647 | Non-empty: 583 | Symbol count: 37

### Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| new | 7 |
| fmt | 5 |
| tick | 5 |
| engage | 4 |
| run_aeb | 2 |
| run_acc | 2 |
| run_lka | 2 |
| run_tja | 2 |
| update_path | 2 |
| safety_check | 2 |
| default | 1 |
| noise | 1 |
| push | 1 |
| disengage | 1 |
| toggle_lka | 1 |
| set_acc_speed | 1 |
| set_headway | 1 |
| disengaged_controller_outputs_none_and_zero_state | 1 |
| engage_enables_l2_features | 1 |
| engage_from_standstill_requests_positive_throttle_on_clear_road | 1 |
| no_lead_then_close_lead_transitions_to_brake | 1 |

### Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |

```
|---:|---|---|---|
| 6 | enum | SaeLevel | data/schema coupling and match/serde break risk |
| 15 | impl | std::fmt::Display for SaeLevel | method behavior and trait semantics |
| 16 | fn | fmt | API and behavior coupling to callers/implementers |
| 30 | enum | FeatureState | data/schema coupling and match/serde break risk |
| 38 | impl | std::fmt::Display for FeatureState | method behavior and trait semantics |
| 39 | fn | fmt | API and behavior coupling to callers/implementers |
| 52 | struct | LaneInfo | data/schema coupling and match/serde break risk |
| 66 | enum | LaneMarkType | data/schema coupling and match/serde break risk |
| 74 | impl | std::fmt::Display for LaneMarkType | method behavior and trait semantics |
| 75 | fn | fmt | API and behavior coupling to callers/implementers |
| 89 | struct | Waypoint | data/schema coupling and match/serde break risk |
| 97 | struct | AutonomousController | data/schema coupling and match/serde break risk |
| 158 | enum | LcaDirection | data/schema coupling and match/serde break risk |
| 165 | enum | SafetyStatus | data/schema coupling and match/serde break risk |
| 172 | impl | std::fmt::Display for SafetyStatus | method behavior and trait semantics |
| 173 | fn | fmt | API and behavior coupling to callers/implementers |
| 183 | impl | Default for AutonomousController | method behavior and trait semantics |
| 184 | fn | default | API and behavior coupling to callers/implementers |
| 189 | impl | AutonomousController | method behavior and trait semantics |
| 190 | fn | new | API and behavior coupling to callers/implementers |
| 245 | fn | tick | API and behavior coupling to callers/implementers |
| 342 | fn | run_aeb | API and behavior coupling to callers/implementers |
| 367 | fn | run_acc | API and behavior coupling to callers/implementers |
| 417 | fn | run_lka | API and behavior coupling to callers/implementers |
| 438 | fn | run_tja | API and behavior coupling to callers/implementers |
| 461 | fn | update_path | API and behavior coupling to callers/implementers |
| 478 | fn | safety_check | API and behavior coupling to callers/implementers |
| 502 | fn | engage | API and behavior coupling to callers/implementers |
| 514 | fn | disengage | API and behavior coupling to callers/implementers |
| 523 | fn | toggle_lka | API and behavior coupling to callers/implementers |
| 531 | fn | set_acc_speed | API and behavior coupling to callers/implementers |
| 534 | fn | set_headway | API and behavior coupling to callers/implementers |
| 540 | module | tests | namespace and compile-time linkage |
| 544 | fn | disengaged_controller_outputs_none_and_zero_state | API and behavior coupling to callers/implementers |
| 569 | fn | engage_enables_l2_features | API and behavior coupling to callers/implementers |
| 580 | fn | engage_from_standstill_requests_positive_throttle_on_clear_road | API and behavior coupling to callers/impl
| 606 | fn | no_lead_then_close_lead_transitions_to_brake | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Autonomous Driving Controller — SAE Level 2/3 AD system. | comment/documentation; changing affects understandi
| 2 | //! Implements: ACC, AEB, LKA, LCA, TJA, Pilot Assist, ISO 26262 ASIL-D safety. | comment/documentation; changing
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | // ■■ SAE Automation Level ■■■
```

```
| 34 |     Override, | support logic; impact depends on neighboring block and data flow. |
| 35 |     Fault, | support logic; impact depends on neighboring block and data flow. |
| 36 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 37 | (blank) | blank separator; no runtime behavior. |
| 38 | impl std::fmt::Display for FeatureState { | implementation binding; behavior semantics and trait conformance live
| 39 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 40 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 41 |             FeatureState::Off => write!(f, "OFF     "), | protocol or I/O critical; changes may impact external i
| 42 |             FeatureState::Ready => write!(f, "READY   "), | protocol or I/O critical; changes may impact externa
| 43 |             FeatureState::Active => write!(f, "ACTIVE  "), | protocol or I/O critical; changes may impact externa
| 44 |             FeatureState::Override => write!(f, "OVERRIDE"), | protocol or I/O critical; changes may impact exte
| 45 |             FeatureState::Fault => write!(f, "FAULT   "), | protocol or I/O critical; changes may impact externa
| 46 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 | // ■■ Lane info ■■■
```

```
| 110 |     pub aeb_brake_cmd: f64, // 0-1 brake command | support logic; impact depends on neighboring block and data f
| 111 |     pub aeb_pre_charge: bool, | support logic; impact depends on neighboring block and data flow. |
| 112 | (blank) | blank separator; no runtime behavior. |
| 113 |     pub lka_state: FeatureState, | support logic; impact depends on neighboring block and data flow. |
| 114 |     pub lka_steer_cmd: f64, // steering torque Nm | support logic; impact depends on neighboring block and data
| 115 | (blank) | blank separator; no runtime behavior. |
| 116 |     pub lca_state: FeatureState, // Lane Change Assist | support logic; impact depends on neighboring block and
| 117 |     pub lca_direction: LcaDirection, | support logic; impact depends on neighboring block and data flow. |
| 118 | (blank) | blank separator; no runtime behavior. |
| 119 |     pub tja_state: FeatureState, // Traffic Jam Assist (stop-and-go) | support logic; impact depends on neighbo
| 120 |     pub tja_stopped_timer: f64, | support logic; impact depends on neighboring block and data flow. |
| 121 | (blank) | blank separator; no runtime behavior. |
| 122 |     pub pilot_state: FeatureState, // Highway Pilot (L3) | support logic; impact depends on neighboring block a
| 123 | (blank) | blank separator; no runtime behavior. |
| 124 |     pub bsm_left: bool, | support logic; impact depends on neighboring block and data flow. |
| 125 |     pub bsm_right: bool, | support logic; impact depends on neighboring block and data flow. |
| 126 | (blank) | blank separator; no runtime behavior. |
| 127 |     // ■ Sensor inputs ■■■
```

```
| 186 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 187 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 188 | (blank) | blank separator; no runtime behavior. |
| 189 | impl AutonomousController { | implementation binding; behavior semantics and trait conformance live here. |
| 190 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 191 |         AutonomousController { | support logic; impact depends on neighboring block and data flow. |
| 192 |             sae_level: SaeLevel::L2, | support logic; impact depends on neighboring block and data flow. |
| 193 |             engaged: false, | support logic; impact depends on neighboring block and data flow. |
| 194 |             driver_hands_on: true, | support logic; impact depends on neighboring block and data flow. |
| 195 |             acc_state: FeatureState::Off, | support logic; impact depends on neighboring block and data flow. |
| 196 |             acc_set_speed_kmh: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 197 |             acc_headway_s: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 198 |             acc_speed_error_integral: 0.0, | support logic; impact depends on neighboring block and data flow.
| 199 |             aeb_state: FeatureState::Ready, | support logic; impact depends on neighboring block and data flow.
| 200 |             aeb_brake_cmd: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 201 |             aeb_pre_charge: false, | support logic; impact depends on neighboring block and data flow. |
| 202 |             lka_state: FeatureState::Off, | support logic; impact depends on neighboring block and data flow. |
| 203 |             lka_steer_cmd: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 204 |             lca_state: FeatureState::Off, | support logic; impact depends on neighboring block and data flow. |
| 205 |             lca_direction: LcaDirection::None, | support logic; impact depends on neighboring block and data flo
| 206 |             tja_state: FeatureState::Off, | support logic; impact depends on neighboring block and data flow. |
| 207 |             tja_stopped_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 208 |             pilot_state: FeatureState::Off, | support logic; impact depends on neighboring block and data flow.
| 209 |             bsm_left: false, | support logic; impact depends on neighboring block and data flow. |
| 210 |             bsm_right: false, | support logic; impact depends on neighboring block and data flow. |
| 211 |             lane: LaneInfo { | support logic; impact depends on neighboring block and data flow. |
| 212 |                 detected: true, | support logic; impact depends on neighboring block and data flow. |
| 213 |                 confidence: 0.95, | support logic; impact depends on neighboring block and data flow. |
| 214 |                 offset_m: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 215 |                 heading_error_deg: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 216 |                 curvature_1_m: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 217 |                 lane_width_m: 3.5, | support logic; impact depends on neighboring block and data flow. |
| 218 |                 left_type: LaneMarkType::Dashed, | support logic; impact depends on neighboring block and data f
| 219 |                 right_type: LaneMarkType::Solid, | support logic; impact depends on neighboring block and data f
| 220 |                 adjacent_left: true, | support logic; impact depends on neighboring block and data flow. |
| 221 |                 adjacent_right: false, | support logic; impact depends on neighboring block and data flow. |
| 222 |             }, | support logic; impact depends on neighboring block and data flow. |
| 223 |             lead_range_m: f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 224 |             lead_speed_ms: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 225 |             ttc_s: f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 226 |             thw_s: f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 227 |             throttle_cmd: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 228 |             brake_cmd: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 229 |             steer_cmd_deg: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 230 |             planned_path: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 231 |             path_valid: false, | support logic; impact depends on neighboring block and data flow. |
| 232 |             driver_alert: false, | support logic; impact depends on neighboring block and data flow. |
| 233 |             hands_off_s: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 234 |             drowsiness_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 235 |             safety_status: SafetyStatus::Normal, | support logic; impact depends on neighboring block and data f
| 236 |             degrade_reason: None, | support logic; impact depends on neighboring block and data flow. |
| 237 |             noise_t: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 238 |             lka_integral: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 239 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 240 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 241 | (blank) | blank separator; no runtime behavior. |
| 242 |     /// Master AD controller tick. | comment/documentation; changing affects understanding and intent traceabil
| 243 |     /// Returns (throttle, brake, steer) overrides — None if no override. | comment/documentation; changing affe
| 244 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent trac
| 245 |     pub fn tick( | function signature/body; changes affect API behavior and control-flow. |
| 246 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 247 |         ego_speed_kmh: f64, | support logic; impact depends on neighboring block and data flow. |
| 248 |         _user_throttle: f64, | support logic; impact depends on neighboring block and data flow. |
| 249 |         _user_brake: f64, | support logic; impact depends on neighboring block and data flow. |
| 250 |         lead_range_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 251 |         lead_speed_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 252 |         lane_offset_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 253 |         lane_heading_err: f64, | support logic; impact depends on neighboring block and data flow. |
| 254 |         lane_confidence: f64, | support logic; impact depends on neighboring block and data flow. |
| 255 |         ttc_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 256 |         bsm_left: bool, | support logic; impact depends on neighboring block and data flow. |
| 257 |         bsm_right: bool, | support logic; impact depends on neighboring block and data flow. |
| 258 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 259 |     ) -> (Option<f64>, Option<f64>, Option<f64>) { | support logic; impact depends on neighboring block and data
| 260 |         self.noise_t += dt; | support logic; impact depends on neighboring block and data flow. |
| 261 |         let ego_speed_ms = ego_speed_kmh / 3.6; | support logic; impact depends on neighboring block and data f
```

```
| 262 | (blank) | blank separator; no runtime behavior. |
| 263 |         // Update inputs | comment/documentation; changing affects understanding and intent traceability. |
| 264 |         self.lead_range_m = lead_range_m; | support logic; impact depends on neighboring block and data flow. |
| 265 |         self.lead_speed_ms = lead_speed_ms; | support logic; impact depends on neighboring block and data flow.
| 266 |         self.ttc_s = ttc_s; | support logic; impact depends on neighboring block and data flow. |
| 267 |         self.thw_s = if ego_speed_ms > 0.1 { | support logic; impact depends on neighboring block and data flow
| 268 |             lead_range_m / ego_speed_ms | support logic; impact depends on neighboring block and data flow. |
| 269 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 270 |             f64::INFINITY | support logic; impact depends on neighboring block and data flow. |
| 271 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 272 |         self.bsm_left = bsm_left; | support logic; impact depends on neighboring block and data flow. |
| 273 |         self.bsm_right = bsm_right; | support logic; impact depends on neighboring block and data flow. |
| 274 |         self.lane.offset_m = lane_offset_m; | support logic; impact depends on neighboring block and data flow.
| 275 |         self.lane.heading_error_deg = lane_heading_err; | support logic; impact depends on neighboring block an
| 276 |         self.lane.confidence = lane_confidence; | support logic; impact depends on neighboring block and data f
| 277 |         self.lane.detected = lane_confidence > 0.4; | support logic; impact depends on neighboring block and da
| 278 | (blank) | blank separator; no runtime behavior. |
| 279 |         // Driver hands-off timer | comment/documentation; changing affects understanding and intent traceabili
| 280 |         if self.driver_hands_on { | runtime gate; condition changes can enable/disable critical paths. |
| 281 |             self.hands_off_s = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 282 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 283 |             self.hands_off_s += dt; | support logic; impact depends on neighboring block and data flow. |
| 284 |             self.driver_alert = self.hands_off_s > 10.0; | support logic; impact depends on neighboring block a
| 285 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 286 | (blank) | blank separator; no runtime behavior. |
| 287 |         // Drowsiness simulation | comment/documentation; changing affects understanding and intent traceability
| 288 |         let n = self.noise_t; | support logic; impact depends on neighboring block and data flow. |
| 289 |         let noise = \|s: f64\| ((s * 127.1 + 311.7).sin() * 43758.5 % 1.0 - 0.5) * 2.0; | support logic; impact
| 290 |         self.drowsiness_pct = (self.drowsiness_pct + noise(n * 0.001) * 0.01).clamp(0.0, 100.0); | support logic
| 291 | (blank) | blank separator; no runtime behavior. |
| 292 |         if !self.engaged { | runtime gate; condition changes can enable/disable critical paths. |
| 293 |             self.throttle_cmd = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 294 |             self.brake_cmd = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 295 |             self.steer_cmd_deg = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 296 |             return (None, None, None); | support logic; impact depends on neighboring block and data flow. |
| 297 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 298 | (blank) | blank separator; no runtime behavior. |
| 299 |         let mut thr: Option<f64> = None; | support logic; impact depends on neighboring block and data flow. |
| 300 |         let mut brk: Option<f64> = None; | support logic; impact depends on neighboring block and data flow. |
| 301 |         let mut str: Option<f64> = None; | support logic; impact depends on neighboring block and data flow. |
| 302 | (blank) | blank separator; no runtime behavior. |
| 303 |         // ■■ AEB — ALWAYS active regardless of engagement ■■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentation
| 304 |         self.run_aeb(ego_speed_ms, ttc_s, &mut brk); | support logic; impact depends on neighboring block and da
| 305 | (blank) | blank separator; no runtime behavior. |
| 306 |         // ■■ ACC — Adaptive Cruise Control ■■■■■■■■■■■■■■■■■■■
```

```
| 338 |         (thr, brk, str) | support logic; impact depends on neighboring block and data flow. |
| 339 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 | (blank) | blank separator; no runtime behavior. |
| 341 |     // ################################################################################ | comment/d
| 342 |     fn run_aeb(&mut self, speed_ms: f64, ttc_s: f64, brk: &mut Option<f64>) { | function signature/body; changes
| 343 |         self.aeb_state = if speed_ms > 2.0 && ttc_s < 2.8 { | support logic; impact depends on neighboring bloc
| 344 |             FeatureState::Active | support logic; impact depends on neighboring block and data flow. |
| 345 |         } else if speed_ms > 1.0 { | support logic; impact depends on neighboring block and data flow. |
| 346 |             FeatureState::Ready | support logic; impact depends on neighboring block and data flow. |
| 347 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 348 |             FeatureState::Off | support logic; impact depends on neighboring block and data flow. |
| 349 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 350 | (blank) | blank separator; no runtime behavior. |
| 351 |         self.aeb_pre_charge = ttc_s < 2.8 && speed_ms > 2.0; | support logic; impact depends on neighboring bloc
| 352 |         self.aeb_brake_cmd = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 353 | (blank) | blank separator; no runtime behavior. |
| 354 |         if ttc_s < 2.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 355 |             // Partial braking phase | comment/documentation; changing affects understanding and intent traceabi
| 356 |             let partial = ((2.5 - ttc_s) / 1.5).clamp(0.0, 0.6); | support logic; impact depends on neighboring
| 357 |             self.aeb_brake_cmd = partial; | support logic; impact depends on neighboring block and data flow. |
| 358 |             *brk = Some(partial.max(brk.unwrap_or(0.0))); | support logic; impact depends on neighboring block a
| 359 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 360 |         if ttc_s < 1.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 361 |             // Full emergency braking | comment/documentation; changing affects understanding and intent traceal
| 362 |             self.aeb_brake_cmd = 1.0; | support logic; impact depends on neighboring block and data flow. |
| 363 |             *brk = Some(1.0); | support logic; impact depends on neighboring block and data flow. |
| 364 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 365 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 366 | (blank) | blank separator; no runtime behavior. |
| 367 |     fn run_acc( | function signature/body; changes affect API behavior and control-flow. |
| 368 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 369 |         speed_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 370 |         lead_range: f64, | support logic; impact depends on neighboring block and data flow. |
| 371 |         _lead_speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 372 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 373 |         thr: &mut Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 374 |         brk: &mut Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 375 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 376 |         let set_speed = self.acc_set_speed_kmh / 3.6; | support logic; impact depends on neighboring block and d
| 377 | (blank) | blank separator; no runtime behavior. |
| 378 |         // Target speed: minimum of set speed and speed needed to maintain headway | comment/documentation; char
| 379 |         let safe_dist = speed_ms * self.acc_headway_s; | support logic; impact depends on neighboring block and
| 380 |         let dist_error = lead_range - safe_dist; | support logic; impact depends on neighboring block and data
| 381 |         let lead_valid = lead_range.is_finite(); | support logic; impact depends on neighboring block and data
| 382 |         let gap_speed = if lead_valid { | support logic; impact depends on neighboring block and data flow. |
| 383 |             (speed_ms + dist_error * 0.5).clamp(0.0, set_speed) | support logic; impact depends on neighboring b
| 384 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 385 |             set_speed | support logic; impact depends on neighboring block and data flow. |
| 386 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 387 |         let target_speed = gap_speed.min(set_speed); | support logic; impact depends on neighboring block and da
| 388 | (blank) | blank separator; no runtime behavior. |
| 389 |         // PI speed controller with anti-windup. | comment/documentation; changing affects understanding and int
| 390 |         let speed_error = target_speed - speed_ms; | support logic; impact depends on neighboring block and data
| 391 |         if !lead_valid { | runtime gate; condition changes can enable/disable critical paths. |
| 392 |             // Prevent stale integral from causing surge when a lead target reappears. | comment/documentation;
| 393 |             self.acc_speed_error_integral *= 0.92; | support logic; impact depends on neighboring block and data
| 394 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 395 | (blank) | blank separator; no runtime behavior. |
| 396 |         let candidate_integral = | support logic; impact depends on neighboring block and data flow. |
| 397 |             (self.acc_speed_error_integral + speed_error * dt).clamp(-20.0, 20.0); | support logic; impact depe
| 398 |         let unsat_control = speed_error * 0.08 + candidate_integral * 0.004; | support logic; impact depends on
| 399 |         let control = unsat_control.clamp(-0.8, 1.0); | support logic; impact depends on neighboring block and d
| 400 | (blank) | blank separator; no runtime behavior. |
| 401 |         // Integrate only when not saturating in the same error direction. | comment/documentation; changing af
| 402 |         let same_direction_saturation = (unsat_control - control).abs() > f64::EPSILON | support logic; impact d
| 403 |             && speed_error.abs() > 0.001 | support logic; impact depends on neighboring block and data flow. |
| 404 |             && speed_error.signum() == control.signum(); | support logic; impact depends on neighboring block a
| 405 |         if !same_direction_saturation { | runtime gate; condition changes can enable/disable critical paths. |
| 406 |             self.acc_speed_error_integral = candidate_integral; | support logic; impact depends on neighboring b
| 407 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 408 | (blank) | blank separator; no runtime behavior. |
| 409 |         if control > 0.02 { | runtime gate; condition changes can enable/disable critical paths. |
| 410 |             *thr = Some(control.clamp(0.0, 1.0)); | support logic; impact depends on neighboring block and data
| 411 |         } else if control < -0.02 { | support logic; impact depends on neighboring block and data flow. |
| 412 |             *brk = Some((-control).clamp(0.0, 0.8).max(brk.unwrap_or(0.0))); | support logic; impact depends on
| 413 |             *thr = Some(0.0); | support logic; impact depends on neighboring block and data flow. |
```

```
| 414 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 415 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 416 | (blank) | blank separator; no runtime behavior. |
| 417 |     fn run_lka( | function signature/body; changes affect API behavior and control-flow. |
| 418 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 419 |         offset: f64, | support logic; impact depends on neighboring block and data flow. |
| 420 |         heading_err: f64, | support logic; impact depends on neighboring block and data flow. |
| 421 |         speed_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 422 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 423 |         str: &mut Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 424 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 425 |         if speed_ms < 3.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 426 |             return; | support logic; impact depends on neighboring block and data flow. |
| 427 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 428 |         // PD controller for lane centering | comment/documentation; changing affects understanding and intent t
| 429 |         let kp = 4.0 * (speed_ms / 30.0).min(1.0); | support logic; impact depends on neighboring block and data
| 430 |         let ki = 0.3; | support logic; impact depends on neighboring block and data flow. |
| 431 |         self.lka_integral += offset * dt; | support logic; impact depends on neighboring block and data flow. |
| 432 |         self.lka_integral = self.lka_integral.clamp(-5.0, 5.0); | support logic; impact depends on neighboring b
| 433 |         let steer = -(kp * offset + ki * self.lka_integral + 0.5 * heading_err); | support logic; impact depends
| 434 |         self.lka_steer_cmd = steer.clamp(-6.0, 6.0); | support logic; impact depends on neighboring block and da
| 435 |         *str = Some(self.lka_steer_cmd); | support logic; impact depends on neighboring block and data flow. |
| 436 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 437 | (blank) | blank separator; no runtime behavior. |
| 438 |     fn run_tja( | function signature/body; changes affect API behavior and control-flow. |
| 439 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 440 |         speed_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 441 |         lead_range: f64, | support logic; impact depends on neighboring block and data flow. |
| 442 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 443 |         thr: &mut Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 444 |         brk: &mut Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 445 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 446 |         // Stop-and-go: if stopped for >3s, wait for lead vehicle to move | comment/documentation; changing affe
| 447 |         if speed_ms < 0.2 { | runtime gate; condition changes can enable/disable critical paths. |
| 448 |             self.tja_stopped_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 449 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 450 |             self.tja_stopped_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 451 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 452 |         // If lead vehicle moved away and we were stopped | comment/documentation; changing affects understandi
| 453 |         if self.tja_stopped_timer > 3.0 && lead_range > 5.0 { | runtime gate; condition changes can enable/disal
| 454 |             *thr = Some(0.2); // gentle pull-away | support logic; impact depends on neighboring block and data
| 455 |         } else if lead_range < 3.0 { | support logic; impact depends on neighboring block and data flow. |
| 456 |             *brk = Some(1.0); | support logic; impact depends on neighboring block and data flow. |
| 457 |             *thr = Some(0.0); | support logic; impact depends on neighboring block and data flow. |
| 458 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 459 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 460 | (blank) | blank separator; no runtime behavior. |
| 461 |     fn update_path(&mut self, speed_ms: f64) { | function signature/body; changes affect API behavior and contro
| 462 |         // Generate simple straight-ahead path (full planning would use A* + splines) | comment/documentation; c
| 463 |         self.planned_path.clear(); | support logic; impact depends on neighboring block and data flow. |
| 464 |         let preview_dist = (speed_ms * 3.5).max(30.0); | support logic; impact depends on neighboring block and
| 465 |         let n_points = 20; | support logic; impact depends on neighboring block and data flow. |
| 466 |         for i in 0..n_points { | iteration logic; may alter timing, throughput, and safety constraints. |
| 467 |             let t = i as f64 / n_points as f64; | support logic; impact depends on neighboring block and data fl
| 468 |             self.planned_path.push(Waypoint { | support logic; impact depends on neighboring block and data flow
| 469 |                 x: t * preview_dist, | support logic; impact depends on neighboring block and data flow. |
| 470 |                 y: -self.lane.offset_m * (1.0 - t), // converge to centre | support logic; impact depends on nei
| 471 |                 speed_ms, | support logic; impact depends on neighboring block and data flow. |
| 472 |                 curvature: self.lane.curvature_1_m, | support logic; impact depends on neighboring block and dat
| 473 |             }); | support logic; impact depends on neighboring block and data flow. |
| 474 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 475 |         self.path_valid = self.lane.detected; | support logic; impact depends on neighboring block and data flow
| 476 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 477 | (blank) | blank separator; no runtime behavior. |
| 478 |     fn safety_check(&mut self, speed_ms: f64, thr: Option<f64>, brk: Option<f64>) { | function signature/body; c
| 479 |         // ISO 26262 ASIL-D plausibility checks | comment/documentation; changing affects understanding and inte
| 480 |         if let (Some(t), Some(b)) = (thr, brk) { | runtime gate; condition changes can enable/disable critical p
| 481 |             if t > 0.1 && b > 0.1 { | runtime gate; condition changes can enable/disable critical paths. |
| 482 |                 // Simultaneous throttle and brake — contradiction! | comment/documentation; changing affects un
| 483 |                 self.safety_status = SafetyStatus::Degraded; | support logic; impact depends on neighboring bloc
| 484 |                 self.degrade_reason = Some("Throttle+Brake simultaneous"); | support logic; impact depends on ne
| 485 |                 return; | support logic; impact depends on neighboring block and data flow. |
| 486 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 487 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 488 |         if speed_ms > 50.0 | runtime gate; condition changes can enable/disable critical paths. |
| 489 |             && matches!(self.acc_state, FeatureState::Off) | support logic; impact depends on neighboring block
```

```
| 490 |             && matches!(self.lka_state, FeatureState::Active) | support logic; impact depends on neighboring blo
| 491 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 492 |             // LKA without ACC at high speed — degrade | comment/documentation; changing affects understanding a
| 493 |             self.safety_status = SafetyStatus::Degraded; | support logic; impact depends on neighboring block an
| 494 |             self.degrade_reason = Some("LKA w/o ACC at high speed"); | support logic; impact depends on neighbor
| 495 |             return; | support logic; impact depends on neighboring block and data flow. |
| 496 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 497 |         self.safety_status = SafetyStatus::Normal; | support logic; impact depends on neighboring block and data
| 498 |         self.degrade_reason = None; | support logic; impact depends on neighboring block and data flow. |
| 499 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 500 | (blank) | blank separator; no runtime behavior. |
| 501 |     // ■ Public control helpers ■■■
```

```
| 566 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 567 | (blank) | blank separator; no runtime behavior. |
| 568 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 569 |     fn engage_enables_l2_features() { | function signature/body; changes affect API behavior and control-flow.
| 570 |         let mut ad = AutonomousController::new(); | support logic; impact depends on neighboring block and data
| 571 |         ad.engage(42.0); | support logic; impact depends on neighboring block and data flow. |
| 572 |         assert!(ad.engaged); | support logic; impact depends on neighboring block and data flow. |
| 573 |         assert_eq!(ad.acc_state, FeatureState::Active); | support logic; impact depends on neighboring block an
| 574 |         assert_eq!(ad.lka_state, FeatureState::Active); | support logic; impact depends on neighboring block an
| 575 |         assert_eq!(ad.tja_state, FeatureState::Active); | support logic; impact depends on neighboring block an
| 576 |         assert!(ad.acc_set_speed_kmh >= 30.0); | support logic; impact depends on neighboring block and data fl
| 577 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 578 | (blank) | blank separator; no runtime behavior. |
| 579 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 580 |     fn engage_from_standstill_requests_positive_throttle_on_clear_road() { | function signature/body; changes a
| 581 |         let mut ad = AutonomousController::new(); | support logic; impact depends on neighboring block and data
| 582 |         ad.engage(0.0); | support logic; impact depends on neighboring block and data flow. |
| 583 | (blank) | blank separator; no runtime behavior. |
| 584 |         let (thr, brk, steer) = ad.tick( | support logic; impact depends on neighboring block and data flow. |
| 585 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 586 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 587 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 588 |             f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 589 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 590 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 591 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 592 |             0.95, | support logic; impact depends on neighboring block and data flow. |
| 593 |             f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 594 |             false, | support logic; impact depends on neighboring block and data flow. |
| 595 |             false, | support logic; impact depends on neighboring block and data flow. |
| 596 |             0.05, | support logic; impact depends on neighboring block and data flow. |
| 597 |         ); | support logic; impact depends on neighboring block and data flow. |
| 598 | (blank) | blank separator; no runtime behavior. |
| 599 |         assert!(thr.is_some_and(\|v\| v > 0.05)); | support logic; impact depends on neighboring block and data
| 600 |         assert!(brk.is_none() \|\| brk == Some(0.0)); | support logic; impact depends on neighboring block and
| 601 |         assert!(steer.is_none() \|\| steer == Some(0.0)); | support logic; impact depends on neighboring block
| 602 |         assert!(ad.throttle_cmd > 0.05); | support logic; impact depends on neighboring block and data flow. |
| 603 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 604 | (blank) | blank separator; no runtime behavior. |
| 605 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 606 |     fn no_lead_then_close_lead_transitions_to_brake() { | function signature/body; changes affect API behavior
| 607 |         let mut ad = AutonomousController::new(); | support logic; impact depends on neighboring block and data
| 608 |         ad.engage(50.0); | support logic; impact depends on neighboring block and data flow. |
| 609 | (blank) | blank separator; no runtime behavior. |
| 610 |         // Build integral/throttle while road is clear. | comment/documentation; changing affects understanding
| 611 |         for _ in 0..120 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 612 |             let _ = ad.tick( | support logic; impact depends on neighboring block and data flow. |
| 613 |                 30.0, | support logic; impact depends on neighboring block and data flow. |
| 614 |                 0.0, | support logic; impact depends on neighboring block and data flow. |
| 615 |                 0.0, | support logic; impact depends on neighboring block and data flow. |
| 616 |                 f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 617 |                 0.0, | support logic; impact depends on neighboring block and data flow. |
| 618 |                 0.0, | support logic; impact depends on neighboring block and data flow. |
| 619 |                 0.0, | support logic; impact depends on neighboring block and data flow. |
| 620 |                 0.95, | support logic; impact depends on neighboring block and data flow. |
| 621 |                 f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 622 |                 false, | support logic; impact depends on neighboring block and data flow. |
| 623 |                 false, | support logic; impact depends on neighboring block and data flow. |
| 624 |                 0.02, | support logic; impact depends on neighboring block and data flow. |
| 625 |             ); | support logic; impact depends on neighboring block and data flow. |
| 626 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 627 |         assert!(ad.throttle_cmd >= 0.0); | support logic; impact depends on neighboring block and data flow. |
| 628 | (blank) | blank separator; no runtime behavior. |
| 629 |         // Sudden close lead should force braking and suppress throttle. | comment/documentation; changing affe
| 630 |         let _ = ad.tick( | support logic; impact depends on neighboring block and data flow. |
| 631 |             30.0, | support logic; impact depends on neighboring block and data flow. |
| 632 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 633 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 634 |             8.0, | support logic; impact depends on neighboring block and data flow. |
| 635 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 636 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 637 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 638 |             0.95, | support logic; impact depends on neighboring block and data flow. |
| 639 |             1.2, | support logic; impact depends on neighboring block and data flow. |
| 640 |             false, | support logic; impact depends on neighboring block and data flow. |
| 641 |             false, | support logic; impact depends on neighboring block and data flow. |
```

```
| 642 |             0.02, | support logic; impact depends on neighboring block and data flow. |
| 643 |         ); | support logic; impact depends on neighboring block and data flow. |
| 644 |         assert!(ad.brake_cmd > 0.0); | support logic; impact depends on neighboring block and data flow. |
| 645 |         assert!(ad.throttle_cmd <= 0.05); | support logic; impact depends on neighboring block and data flow. |
| 646 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 647 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/bin/cat_comm_bridge.rs

Role: operational executable entrypoint

Line count: 367 | Non-empty: 347 | Symbol count: 15

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| push_serial | 17 |
| capabilities | 10 |
| did_ascii | 7 |
| get | 6 |
| write_response | 5 |
| make_ok | 4 |
| default | 2 |
| now_ms | 2 |
| handle_uds_serial_request | 2 |
| flush | 2 |
| new | 1 |
| is_empty | 1 |
| len | 1 |
| main | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 9 | enum | BridgeRequest | data/schema coupling and match/serde break risk |
| 29 | struct | BridgeDiagPayload | data/schema coupling and match/serde break risk |
| 38 | struct | BridgeResponse | data/schema coupling and match/serde break risk |
| 49 | struct | BridgeState | data/schema coupling and match/serde break risk |
| 58 | impl | Default for BridgeState | method behavior and trait semantics |
| 59 | fn | default | API and behavior coupling to callers/implementers |
| 71 | impl | BridgeState | method behavior and trait semantics |
| 72 | fn | capabilities | API and behavior coupling to callers/implementers |
| 81 | fn | make_ok | API and behavior coupling to callers/implementers |
| 93 | fn | push_serial | API and behavior coupling to callers/implementers |
| 101 | fn | handle_uds_serial_request | API and behavior coupling to callers/implementers |
| 168 | fn | did_ascii | API and behavior coupling to callers/implementers |
| 175 | fn | main | API and behavior coupling to callers/implementers |
| 345 | fn | write_response | API and behavior coupling to callers/implementers |
| 362 | fn | now_ms | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::io::{self, BufRead, Write}; | import wiring; changing path/name can break compilation and module linkage
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use auto_breaking::io::hw::{Frame, SerialFrame}; | import wiring; changing path/name can break compilation and mo
| 5 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | #[derive(Debug, Deserialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 8 | #[serde(tag = "cmd", rename_all = "snake_case")] | comment/documentation; changing affects understanding and inte
| 9 | enum BridgeRequest { | state machine variants; changes ripple through matches and business logic. |
| 10 |     Init { | support logic; impact depends on neighboring block and data flow. |
| 11 |         dry_run: bool, | support logic; impact depends on neighboring block and data flow. |
| 12 |         enable_write: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 13 |         can_interface: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 14 |         serial_port: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 15 |         template_dir: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 16 |     }, | support logic; impact depends on neighboring block and data flow. |
| 17 |     Read { | support logic; impact depends on neighboring block and data flow. |
```

```
| 18 |         timeout_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 19 |     }, | support logic; impact depends on neighboring block and data flow. |
| 20 |     TryRead, | support logic; impact depends on neighboring block and data flow. |
| 21 |     Send { | support logic; impact depends on neighboring block and data flow. |
| 22 |         frame: Frame, | support logic; impact depends on neighboring block and data flow. |
| 23 |     }, | support logic; impact depends on neighboring block and data flow. |
| 24 |     Close, | support logic; impact depends on neighboring block and data flow. |
| 25 |     Ping, | support logic; impact depends on neighboring block and data flow. |
| 26 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 | (blank) | blank separator; no runtime behavior. |
| 28 | #[derive(Debug, Serialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 29 | struct BridgeDiagPayload { | data layout contract; field changes can break callers/serde/tests. |
| 30 |     wait_frame_count_delta: u32, | support logic; impact depends on neighboring block and data flow. |
| 31 |     sequence_error_count_delta: u32, | support logic; impact depends on neighboring block and data flow. |
| 32 |     flowcontrol_timeout_count_delta: u32, | support logic; impact depends on neighboring block and data flow. |
| 33 |     fc_blocksize: u8, | support logic; impact depends on neighboring block and data flow. |
| 34 |     fc_stmin_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 35 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 36 | (blank) | blank separator; no runtime behavior. |
| 37 | #[derive(Debug, Serialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 38 | struct BridgeResponse { | data layout contract; field changes can break callers/serde/tests. |
| 39 |     ok: bool, | support logic; impact depends on neighboring block and data flow. |
| 40 |     frame: Option<Frame>, | support logic; impact depends on neighboring block and data flow. |
| 41 |     error: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 42 |     kind: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 43 |     protocol_version: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 44 |     capabilities: Option<Vec<String>>, | support logic; impact depends on neighboring block and data flow. |
| 45 |     diag: Option<BridgeDiagPayload>, | support logic; impact depends on neighboring block and data flow. |
| 46 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 | (blank) | blank separator; no runtime behavior. |
| 48 | #[derive(Debug)] | comment/documentation; changing affects understanding and intent traceability. |
| 49 | struct BridgeState { | data layout contract; field changes can break callers/serde/tests. |
| 50 |     rx_queue: VecDeque<Frame>, | support logic; impact depends on neighboring block and data flow. |
| 51 |     dry_run: bool, | support logic; impact depends on neighboring block and data flow. |
| 52 |     enable_write: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 53 |     initialized: bool, | support logic; impact depends on neighboring block and data flow. |
| 54 |     negotiated_stmin_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 55 |     negotiated_blocksize: u8, | support logic; impact depends on neighboring block and data flow. |
| 56 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 57 | (blank) | blank separator; no runtime behavior. |
| 58 | impl Default for BridgeState { | implementation binding; behavior semantics and trait conformance live here. |
| 59 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 60 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 61 |             rx_queue: VecDeque::new(), | support logic; impact depends on neighboring block and data flow. |
| 62 |             dry_run: true, | support logic; impact depends on neighboring block and data flow. |
| 63 |             enable_write: false, | protocol or I/O critical; changes may impact external integration correctness
| 64 |             initialized: false, | support logic; impact depends on neighboring block and data flow. |
| 65 |             negotiated_stmin_ms: 5, | support logic; impact depends on neighboring block and data flow. |
| 66 |             negotiated_blocksize: 8, | support logic; impact depends on neighboring block and data flow. |
| 67 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 68 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 69 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 70 | (blank) | blank separator; no runtime behavior. |
| 71 | impl BridgeState { | implementation binding; behavior semantics and trait conformance live here. |
| 72 |     fn capabilities(&self) -> Vec<String> { | function signature/body; changes affect API behavior and control-f
| 73 |         vec![ | support logic; impact depends on neighboring block and data flow. |
| 74 |             "serial".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 75 |             "can".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 76 |             "write".to_string(), | protocol or I/O critical; changes may impact external integration correctness
| 77 |             "uds_flash".to_string(), | protocol or I/O critical; changes may impact external integration correctn
| 78 |         ] | support logic; impact depends on neighboring block and data flow. |
| 79 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 80 | (blank) | blank separator; no runtime behavior. |
| 81 |     fn make_ok(&self) -> BridgeResponse { | function signature/body; changes affect API behavior and control-flo
| 82 |         BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
| 83 |             ok: true, | support logic; impact depends on neighboring block and data flow. |
| 84 |             frame: None, | support logic; impact depends on neighboring block and data flow. |
| 85 |             error: None, | support logic; impact depends on neighboring block and data flow. |
| 86 |             kind: None, | support logic; impact depends on neighboring block and data flow. |
| 87 |             protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighboring block and d
| 88 |             capabilities: Some(self.capabilities()), | support logic; impact depends on neighboring block and dat
| 89 |             diag: None, | support logic; impact depends on neighboring block and data flow. |
| 90 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 91 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 92 | (blank) | blank separator; no runtime behavior. |
| 93 |     fn push_serial(&mut self, bytes: Vec<u8>, hint: &str) { | function signature/body; changes affect API behavio
```

```
| 94 |         self.rx_queue.push_back(Frame::Serial(SerialFrame { | support logic; impact depends on neighboring block
| 95 |             bytes, | support logic; impact depends on neighboring block and data flow. |
| 96 |             protocol_hint: Some(hint.to_string()), | support logic; impact depends on neighboring block and data
| 97 |             timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 98 |         })); | support logic; impact depends on neighboring block and data flow. |
| 99 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 100 | (blank) | blank separator; no runtime behavior. |
| 101 |     fn handle_uds_serial_request(&mut self, req: &[u8]) { | function signature/body; changes affect API behavio
| 102 |         if req.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 103 |             return; | support logic; impact depends on neighboring block and data flow. |
| 104 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 105 | (blank) | blank separator; no runtime behavior. |
| 106 |         match req[0] { | branch dispatcher; arm changes alter behavior class handling. |
| 107 |             0x10 => { | support logic; impact depends on neighboring block and data flow. |
| 108 |                 let sub = req.get(1).copied().unwrap_or(0x01); | support logic; impact depends on neighboring b
| 109 |                 self.push_serial(vec![0x50, sub, 0x00, 0x32, 0x01, 0xF4], "uds-raw"); | protocol or I/O critica
| 110 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 111 |             0x22 => { | support logic; impact depends on neighboring block and data flow. |
| 112 |                 if req.len() < 3 { | runtime gate; condition changes can enable/disable critical paths. |
| 113 |                     self.push_serial(vec![0x7F, 0x22, 0x13], "uds-raw"); | protocol or I/O critical; changes ma
| 114 |                     return; | support logic; impact depends on neighboring block and data flow. |
| 115 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 116 |                 let did = ((req[1] as u16) << 8) \| req[2] as u16; | support logic; impact depends on neighborin
| 117 |                 match did { | branch dispatcher; arm changes alter behavior class handling. |
| 118 |                     0xF190 => self.did_ascii(did, b"CATVIN1234567890"), | support logic; impact depends on neig
| 119 |                     0xF189 => self.did_ascii(did, b"SW-1.0.0"), | support logic; impact depends on neighboring
| 120 |                     0xF191 => self.did_ascii(did, b"HW-1.0.0"), | support logic; impact depends on neighboring
| 121 |                     0xF180 => self.did_ascii(did, b"CAL-001"), | support logic; impact depends on neighboring b
| 122 |                     0xF18C => self.did_ascii(did, b"ECU-SN-0001"), | support logic; impact depends on neighborin
| 123 |                     0xF184 => self.did_ascii(did, b"FP-TRUSTED-01"), | support logic; impact depends on neighbo
| 124 |                     0xDD0E => { | support logic; impact depends on neighboring block and data flow. |
| 125 |                         // 12.5V with factor 0.05 => raw 250 (0x00FA) | comment/documentation; changing affects
| 126 |                         self.push_serial(vec![0x62, 0xDD, 0x0E, 0x00, 0xFA], "uds-raw"); | protocol or I/O crit
| 127 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 128 |                     _ => self.push_serial(vec![0x7F, 0x22, 0x31], "uds-raw"), | protocol or I/O critical; change
| 129 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 130 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 131 |             0x27 => { | support logic; impact depends on neighboring block and data flow. |
| 132 |                 let sub = req.get(1).copied().unwrap_or(0x00); | support logic; impact depends on neighboring b
| 133 |                 match sub { | branch dispatcher; arm changes alter behavior class handling. |
| 134 |                     0x05 => self.push_serial(vec![0x67, 0x05, 0x12, 0x34, 0x56, 0x78], "uds-raw"), | protocol o
| 135 |                     0x06 => self.push_serial(vec![0x67, 0x06], "uds-raw"), | protocol or I/O critical; changes
| 136 |                     _ => self.push_serial(vec![0x7F, 0x27, 0x12], "uds-raw"), | protocol or I/O critical; change
| 137 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 139 |             0x34 => { | support logic; impact depends on neighboring block and data flow. |
| 140 |                 // Positive response with max block length hint. | comment/documentation; changing affects unde
| 141 |                 self.push_serial(vec![0x74, 0x20, 0x00, 0x40], "uds-raw"); | protocol or I/O critical; changes
| 142 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 143 |             0x36 => { | support logic; impact depends on neighboring block and data flow. |
| 144 |                 let seq = req.get(1).copied().unwrap_or(0x00); | support logic; impact depends on neighboring b
| 145 |                 self.push_serial(vec![0x76, seq], "uds-raw"); | protocol or I/O critical; changes may impact ex
| 146 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 147 |             0x37 => { | support logic; impact depends on neighboring block and data flow. |
| 148 |                 self.push_serial(vec![0x77, 0x00], "uds-raw"); | protocol or I/O critical; changes may impact e
| 149 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 150 |             0x31 => { | support logic; impact depends on neighboring block and data flow. |
| 151 |                 let sub = req.get(1).copied().unwrap_or(0x01); | support logic; impact depends on neighboring b
| 152 |                 let h = req.get(2).copied().unwrap_or(0x00); | support logic; impact depends on neighboring bloc
| 153 |                 let l = req.get(3).copied().unwrap_or(0x00); | support logic; impact depends on neighboring bloc
| 154 |                 self.push_serial(vec![0x71, sub, h, l], "uds-raw"); | protocol or I/O critical; changes may impa
| 155 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 156 |             0x14 => { | support logic; impact depends on neighboring block and data flow. |
| 157 |                 self.push_serial(vec![0x54], "uds-raw"); | protocol or I/O critical; changes may impact externa
| 158 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 159 |             0x3E => { | support logic; impact depends on neighboring block and data flow. |
| 160 |                 self.push_serial(vec![0x7E, 0x00], "uds-raw"); | protocol or I/O critical; changes may impact e
| 161 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 162 |             _ => { | support logic; impact depends on neighboring block and data flow. |
| 163 |                 self.push_serial(vec![0x7F, req[0], 0x11], "uds-raw"); | protocol or I/O critical; changes may
| 164 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 165 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 167 | (blank) | blank separator; no runtime behavior. |
| 168 |     fn did_ascii(&mut self, did: u16, payload: &[u8]) { | function signature/body; changes affect API behavior
| 169 |         let mut rsp = vec![0x62, (did >> 8) as u8, (did & 0xFF) as u8]; | support logic; impact depends on neig
```

```
| 170 |         rsp.extend_from_slice(payload); | support logic; impact depends on neighboring block and data flow. |
| 171 |         self.push_serial(rsp, "uds-raw"); | protocol or I/O critical; changes may impact external integration co
| 172 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 173 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 174 | (blank) | blank separator; no runtime behavior. |
| 175 | fn main() { | function signature/body; changes affect API behavior and control-flow. |
| 176 |     let args: Vec<String> = std::env::args().collect(); | support logic; impact depends on neighboring block and
| 177 |     if !args.iter().any(\|a\| a == "--mode=stdio-jsonl") { | runtime gate; condition changes can enable/disable
| 178 |         eprintln!("cat_comm_bridge: use --mode=stdio-jsonl"); | support logic; impact depends on neighboring bl
| 179 |         std::process::exit(2); | support logic; impact depends on neighboring block and data flow. |
| 180 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 181 | (blank) | blank separator; no runtime behavior. |
| 182 |     let stdin = io::stdin(); | support logic; impact depends on neighboring block and data flow. |
| 183 |     let mut stdout = io::stdout(); | support logic; impact depends on neighboring block and data flow. |
| 184 |     let mut state = BridgeState::default(); | support logic; impact depends on neighboring block and data flow.
| 185 | (blank) | blank separator; no runtime behavior. |
| 186 |     for line in stdin.lock().lines() { | iteration logic; may alter timing, throughput, and safety constraints.
| 187 |         let line = match line { | support logic; impact depends on neighboring block and data flow. |
| 188 |             Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 189 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 190 |                 write_response( | protocol or I/O critical; changes may impact external integration correctness
| 191 |                     &mut stdout, | support logic; impact depends on neighboring block and data flow. |
| 192 |                     BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
| 193 |                         ok: false, | support logic; impact depends on neighboring block and data flow. |
| 194 |                         frame: None, | support logic; impact depends on neighboring block and data flow. |
| 195 |                         error: Some(format!("stdin read failed: {e}")), | support logic; impact depends on neigh
| 196 |                         kind: Some("parse".to_string()), | support logic; impact depends on neighboring block a
| 197 |                         protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighborin
| 198 |                         capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring
| 199 |                         diag: None, | support logic; impact depends on neighboring block and data flow. |
| 200 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 201 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 202 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 203 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 204 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 205 | (blank) | blank separator; no runtime behavior. |
| 206 |         let req = match serde_json::from_str::<BridgeRequest>(line.trim()) { | support logic; impact depends on
| 207 |             Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 208 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 209 |                 write_response( | protocol or I/O critical; changes may impact external integration correctness
| 210 |                     &mut stdout, | support logic; impact depends on neighboring block and data flow. |
| 211 |                     BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
| 212 |                         ok: false, | support logic; impact depends on neighboring block and data flow. |
| 213 |                         frame: None, | support logic; impact depends on neighboring block and data flow. |
| 214 |                         error: Some(format!("invalid request json: {e}")), | support logic; impact depends on n
| 215 |                         kind: Some("parse".to_string()), | support logic; impact depends on neighboring block a
| 216 |                         protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighborin
| 217 |                         capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring
| 218 |                         diag: None, | support logic; impact depends on neighboring block and data flow. |
| 219 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 220 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 221 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 222 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 223 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 224 | (blank) | blank separator; no runtime behavior. |
| 225 |         let rsp = match req { | support logic; impact depends on neighboring block and data flow. |
| 226 |             BridgeRequest::Init { | support logic; impact depends on neighboring block and data flow. |
| 227 |                 dry_run, | support logic; impact depends on neighboring block and data flow. |
| 228 |                 enable_write, | protocol or I/O critical; changes may impact external integration correctness.
| 229 |                 can_interface, | support logic; impact depends on neighboring block and data flow. |
| 230 |                 serial_port, | support logic; impact depends on neighboring block and data flow. |
| 231 |                 template_dir, | support logic; impact depends on neighboring block and data flow. |
| 232 |             } => { | support logic; impact depends on neighboring block and data flow. |
| 233 |                 state.dry_run = dry_run; | support logic; impact depends on neighboring block and data flow. |
| 234 |                 state.enable_write = enable_write; | protocol or I/O critical; changes may impact external inte
| 235 |                 state.initialized = true; | support logic; impact depends on neighboring block and data flow. |
| 236 |                 let _ = (can_interface, serial_port, template_dir); | support logic; impact depends on neighbor
| 237 |                 // Seed one serial frame so detect_ecms has at least one observed channel. | comment/documentati
| 238 |                 state.push_serial(vec![0x62, 0xF1, 0x90, b'O', b'K'], "bridge-heartbeat"); | support logic; impa
| 239 |                 state.make_ok() | support logic; impact depends on neighboring block and data flow. |
| 240 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 241 |             BridgeRequest::Ping => { | support logic; impact depends on neighboring block and data flow. |
| 242 |                 if state.initialized { | runtime gate; condition changes can enable/disable critical paths. |
| 243 |                     state.make_ok() | support logic; impact depends on neighboring block and data flow. |
| 244 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 245 |                     BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
```

```
| 246 |                         ok: false, | support logic; impact depends on neighboring block and data flow. |
| 247 |                         frame: None, | support logic; impact depends on neighboring block and data flow. |
| 248 |                         error: Some("bridge not initialized".to_string()), | support logic; impact depends on ne
| 249 |                         kind: Some("permission".to_string()), | support logic; impact depends on neighboring blo
| 250 |                         protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighborin
| 251 |                         capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring
| 252 |                         diag: None, | support logic; impact depends on neighboring block and data flow. |
| 253 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 254 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 255 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 256 |             BridgeRequest::Read { timeout_ms } => { | support logic; impact depends on neighboring block and da
| 257 |                 let _ = timeout_ms; | support logic; impact depends on neighboring block and data flow. |
| 258 |                 if let Some(frame) = state.rx_queue.pop_front() { | runtime gate; condition changes can enable/d
| 259 |                     BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
| 260 |                         ok: true, | support logic; impact depends on neighboring block and data flow. |
| 261 |                         frame: Some(frame), | support logic; impact depends on neighboring block and data flow.
| 262 |                         error: None, | support logic; impact depends on neighboring block and data flow. |
| 263 |                         kind: None, | support logic; impact depends on neighboring block and data flow. |
| 264 |                         protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighborin
| 265 |                         capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring
| 266 |                         diag: Some(BridgeDiagPayload { | support logic; impact depends on neighboring block and
| 267 |                             wait_frame_count_delta: 0, | support logic; impact depends on neighboring block and
| 268 |                             sequence_error_count_delta: 0, | support logic; impact depends on neighboring block
| 269 |                             flowcontrol_timeout_count_delta: 0, | support logic; impact depends on neighboring b
| 270 |                             fc_blocksize: state.negotiated_blocksize, | support logic; impact depends on neighbo
| 271 |                             fc_stmin_ms: state.negotiated_stmin_ms, | support logic; impact depends on neighbori
| 272 |                         }), | support logic; impact depends on neighboring block and data flow. |
| 273 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 274 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 275 |                     BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
| 276 |                         ok: false, | support logic; impact depends on neighboring block and data flow. |
| 277 |                         frame: None, | support logic; impact depends on neighboring block and data flow. |
| 278 |                         error: Some("read timeout".to_string()), | support logic; impact depends on neighboring
| 279 |                         kind: Some("timeout".to_string()), | support logic; impact depends on neighboring block
| 280 |                         protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighborin
| 281 |                         capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring
| 282 |                         diag: None, | support logic; impact depends on neighboring block and data flow. |
| 283 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 284 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 285 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 286 |             BridgeRequest::TryRead => BridgeResponse { | support logic; impact depends on neighboring block and
| 287 |                 ok: true, | support logic; impact depends on neighboring block and data flow. |
| 288 |                 frame: state.rx_queue.pop_front(), | support logic; impact depends on neighboring block and data
| 289 |                 error: None, | support logic; impact depends on neighboring block and data flow. |
| 290 |                 kind: None, | support logic; impact depends on neighboring block and data flow. |
| 291 |                 protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighboring block
| 292 |                 capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring block a
| 293 |                 diag: None, | support logic; impact depends on neighboring block and data flow. |
| 294 |             }, | support logic; impact depends on neighboring block and data flow. |
| 295 |             BridgeRequest::Send { frame } => { | support logic; impact depends on neighboring block and data flo
| 296 |                 if !state.dry_run && !state.enable_write { | runtime gate; condition changes can enable/disable
| 297 |                     BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
| 298 |                         ok: false, | support logic; impact depends on neighboring block and data flow. |
| 299 |                         frame: None, | support logic; impact depends on neighboring block and data flow. |
| 300 |                         error: Some("write blocked by bridge policy".to_string()), | protocol or I/O critical; c
| 301 |                         kind: Some("write_blocked".to_string()), | protocol or I/O critical; changes may impact
| 302 |                         protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighborin
| 303 |                         capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring
| 304 |                         diag: None, | support logic; impact depends on neighboring block and data flow. |
| 305 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 306 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 307 |                     if let Frame::Serial(sf) = &frame { | runtime gate; condition changes can enable/disable cri
| 308 |                         let is_uds = sf | protocol or I/O critical; changes may impact external integration cor
| 309 |                             .protocol_hint | support logic; impact depends on neighboring block and data flow.
| 310 |                             .as_deref() | support logic; impact depends on neighboring block and data flow. |
| 311 |                             .map(\|h\| h.eq_ignore_ascii_case("uds-raw")) | protocol or I/O critical; changes ma
| 312 |                             .unwrap_or(false); | support logic; impact depends on neighboring block and data flo
| 313 |                         if is_uds { | runtime gate; condition changes can enable/disable critical paths. |
| 314 |                             state.handle_uds_serial_request(&sf.bytes); | protocol or I/O critical; changes may
| 315 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 316 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 317 |                     BridgeResponse { | support logic; impact depends on neighboring block and data flow. |
| 318 |                         ok: true, | support logic; impact depends on neighboring block and data flow. |
| 319 |                         frame: None, | support logic; impact depends on neighboring block and data flow. |
| 320 |                         error: None, | support logic; impact depends on neighboring block and data flow. |
| 321 |                         kind: None, | support logic; impact depends on neighboring block and data flow. |
```

```
| 322 |                        protocol_version: Some("1.0".to_string()), | support logic; impact depends on neighborin
| 323 |                        capabilities: Some(state.capabilities()), | support logic; impact depends on neighboring
| 324 |                        diag: Some(BridgeDiagPayload { | support logic; impact depends on neighboring block and
| 325 |                            wait_frame_count_delta: 0, | support logic; impact depends on neighboring block and
| 326 |                            sequence_error_count_delta: 0, | support logic; impact depends on neighboring block
| 327 |                            flowcontrol_timeout_count_delta: 0, | support logic; impact depends on neighboring b
| 328 |                            fc_blocksize: state.negotiated_blocksize, | support logic; impact depends on neighbo
| 329 |                            fc_stmin_ms: state.negotiated_stmin_ms, | support logic; impact depends on neighbor
| 330 |                        }), | support logic; impact depends on neighboring block and data flow. |
| 331 |                    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 332 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 333 |            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 334 |            BridgeRequest::Close => { | support logic; impact depends on neighboring block and data flow. |
| 335 |                let rsp = state.make_ok(); | support logic; impact depends on neighboring block and data flow.
| 336 |                write_response(&mut stdout, rsp); | protocol or I/O critical; changes may impact external integ
| 337 |                break; | support logic; impact depends on neighboring block and data flow. |
| 338 |            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 339 |        }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 | (blank) | blank separator; no runtime behavior. |
| 341 |        write_response(&mut stdout, rsp); | protocol or I/O critical; changes may impact external integration c
| 342 |    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 343 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 344 | (blank) | blank separator; no runtime behavior. |
| 345 | fn write_response(stdout: &mut io::Stdout, rsp: BridgeResponse) { | function signature/body; changes affect API
| 346 |     let line = match serde_json::to_string(&rsp) { | support logic; impact depends on neighboring block and data
| 347 |         Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 348 |         Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 349 |             let fallback = format!( | support logic; impact depends on neighboring block and data flow. |
| 350 |                 "{{\"ok\":false,\"error\":\"serialize failed: {}\",\"kind\":\"parse\"}}", | support logic; impac
| 351 |                 e | support logic; impact depends on neighboring block and data flow. |
| 352 |             ); | support logic; impact depends on neighboring block and data flow. |
| 353 |             let _ = writeln!(stdout, "{fallback}"); | protocol or I/O critical; changes may impact external inte
| 354 |             let _ = stdout.flush(); | support logic; impact depends on neighboring block and data flow. |
| 355 |             return; | support logic; impact depends on neighboring block and data flow. |
| 356 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 357 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 358 |     let _ = writeln!(stdout, "{line}"); | protocol or I/O critical; changes may impact external integration cor
| 359 |     let _ = stdout.flush(); | support logic; impact depends on neighboring block and data flow. |
| 360 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 361 | (blank) | blank separator; no runtime behavior. |
| 362 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 363 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 364 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data fl
| 365 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 366 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 367 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/bin/simulator_cli.rs

Role: operational executable entrypoint

Line count: 142 | Non-empty: 127 | Symbol count: 4

## Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| has_flag | 6 |
| arg_value | 5 |
| arg_value_prefixed | 5 |
| key_advance | 3 |
| from_cli_args | 2 |
| get | 1 |
| main | 1 |
| probe_live_adapter | 1 |
| capabilities_summary | 1 |
| run_all_phases | 1 |
| new | 1 |
| tick | 1 |
| network_health_score_01 | 1 |
| total_errors | 1 |

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | fn | arg_value | API and behavior coupling to callers/implementers |
| 15 | fn | has_flag | API and behavior coupling to callers/implementers |
| 19 | fn | arg_value_prefixed | API and behavior coupling to callers/implementers |
| 24 | fn | main | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::env; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::path::Path; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | use std::path::PathBuf; | import wiring; changing path/name can break compilation and module linkage. |
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use auto_breaking::HeavyMachinery; | import wiring; changing path/name can break compilation and module linkage.
| 6 | use auto_breaking::io; | import wiring; changing path/name can break compilation and module linkage. |
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | fn arg_value(args: &[String], name: &str) -> Option<String> { | function signature/body; changes affect API behavi
| 9 |     args.iter() | support logic; impact depends on neighboring block and data flow. |
| 10 |         .position(\|a\| a == name) | support logic; impact depends on neighboring block and data flow. |
| 11 |         .and_then(\|i\| args.get(i + 1)) | support logic; impact depends on neighboring block and data flow. |
| 12 |         .cloned() | support logic; impact depends on neighboring block and data flow. |
| 13 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 14 | (blank) | blank separator; no runtime behavior. |
| 15 | fn has_flag(args: &[String], name: &str) -> bool { | function signature/body; changes affect API behavior and co
| 16 |     args.iter().any(\|a\| a == name) | support logic; impact depends on neighboring block and data flow. |
| 17 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 18 | (blank) | blank separator; no runtime behavior. |
| 19 | fn arg_value_prefixed(args: &[String], prefix: &str) -> Option<String> { | function signature/body; changes affe
| 20 |     args.iter() | support logic; impact depends on neighboring block and data flow. |
| 21 |         .find_map(\|a\| a.strip_prefix(prefix).map(\|v\| v.to_string())) | support logic; impact depends on neig
| 22 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | fn main() { | function signature/body; changes affect API behavior and control-flow. |
| 25 |     let args: Vec<String> = env::args().collect(); | support logic; impact depends on neighboring block and data
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 |     if has_flag(&args, "--validate-cat-bridge") { | runtime gate; condition changes can enable/disable critical
| 28 |         let cfg = match io::hw::HwConfig::from_cli_args(&args) { | support logic; impact depends on neighboring
| 29 |             Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 30 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 31 |                 eprintln!("bridge validate config error: {}", e); | support logic; impact depends on neighboring
| 32 |                 std::process::exit(2); | support logic; impact depends on neighboring block and data flow. |
| 33 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 35 | (blank) | blank separator; no runtime behavior. |
| 36 |         match io::hw::probe_live_adapter(&cfg) { | branch dispatcher; arm changes alter behavior class handling.
| 37 |             Ok(info) => { | support logic; impact depends on neighboring block and data flow. |
| 38 |                 println!( | support logic; impact depends on neighboring block and data flow. |
| 39 |                     "bridge_validate=ok mode={:?} capabilities={} adapter_info={}", | error propagation point; c
| 40 |                     cfg.mode, | support logic; impact depends on neighboring block and data flow. |
| 41 |                     cfg.capabilities_summary(), | support logic; impact depends on neighboring block and data fl
| 42 |                     info | support logic; impact depends on neighboring block and data flow. |
| 43 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 44 |                 return; | support logic; impact depends on neighboring block and data flow. |
| 45 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 46 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 47 |                 eprintln!("bridge_validate=failed error={}", e); | support logic; impact depends on neighboring
| 48 |                 std::process::exit(1); | support logic; impact depends on neighboring block and data flow. |
| 49 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 50 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 51 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 | (blank) | blank separator; no runtime behavior. |
| 53 |     if has_flag(&args, "--run-production-phases") { | runtime gate; condition changes can enable/disable critica
| 54 |         let cfg = match io::hw::HwConfig::from_cli_args(&args) { | support logic; impact depends on neighboring
| 55 |             Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 56 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 57 |                 eprintln!("production-phase config error: {}", e); | support logic; impact depends on neighborin
| 58 |                 std::process::exit(2); | support logic; impact depends on neighboring block and data flow. |
| 59 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 61 |         let target_sa = arg_value_prefixed(&args, "--target-sa=") | support logic; impact depends on neighboring
| 62 |             .and_then(\|v\| u8::from_str_radix(v.trim_start_matches("0x"), 16).ok()) | support logic; impact dep
| 63 |             .or_else(\|\| arg_value_prefixed(&args, "--target-sa-dec=").and_then(\|v\| v.parse::<u8>().ok())); |
| 64 |         let firmware = arg_value_prefixed(&args, "--firmware=").map(PathBuf::from); | support logic; impact depe
| 65 |         let report_dir = arg_value_prefixed(&args, "--phase-report-dir=") | support logic; impact depends on nei
```

```
| 66 |             .map(PathBuf::from) | support logic; impact depends on neighboring block and data flow. |
| 67 |             .unwrap_or_else(\|\| PathBuf::from("reports")); | support logic; impact depends on neighboring block
| 68 |         let options = io::production_program::ProgramOptions { | support logic; impact depends on neighboring bl
| 69 |             strict_mode: !has_flag(&args, "--allow-blocked-phases"), | support logic; impact depends on neighbor
| 70 |             execute_flash: has_flag(&args, "--execute-flash"), | protocol or I/O critical; changes may impact ex
| 71 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 72 | (blank) | blank separator; no runtime behavior. |
| 73 |         let result = io::production_program::run_all_phases( | support logic; impact depends on neighboring bloc
| 74 |             &cfg, | support logic; impact depends on neighboring block and data flow. |
| 75 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 76 |             firmware.as_deref().map(Path::new), | support logic; impact depends on neighboring block and data fl
| 77 |             &report_dir, | support logic; impact depends on neighboring block and data flow. |
| 78 |             options, | support logic; impact depends on neighboring block and data flow. |
| 79 |         ); | support logic; impact depends on neighboring block and data flow. |
| 80 |         match result { | branch dispatcher; arm changes alter behavior class handling. |
| 81 |             Ok(path) => { | support logic; impact depends on neighboring block and data flow. |
| 82 |                 println!( | support logic; impact depends on neighboring block and data flow. |
| 83 |                     "production_phases=done report_json={} mode={:?}", | error propagation point; changes alter
| 84 |                     path.display(), | support logic; impact depends on neighboring block and data flow. |
| 85 |                     cfg.mode | support logic; impact depends on neighboring block and data flow. |
| 86 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 87 |                 return; | support logic; impact depends on neighboring block and data flow. |
| 88 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 90 |                 eprintln!("production phases failed: {}", e); | support logic; impact depends on neighboring bloc
| 91 |                 std::process::exit(1); | support logic; impact depends on neighboring block and data flow. |
| 92 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 93 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 94 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 95 | (blank) | blank separator; no runtime behavior. |
| 96 |     let seed = arg_value(&args, "--seed") | support logic; impact depends on neighboring block and data flow. |
| 97 |         .and_then(\|s\| s.parse::<u64>().ok()) | support logic; impact depends on neighboring block and data flow
| 98 |         .unwrap_or(1); | support logic; impact depends on neighboring block and data flow. |
| 99 |     let model = arg_value(&args, "--model").unwrap_or_else(\|\| "reduced".to_string()); | support logic; impact
| 100 |     let steps = arg_value(&args, "--steps") | support logic; impact depends on neighboring block and data flow.
| 101 |         .and_then(\|s\| s.parse::<usize>().ok()) | support logic; impact depends on neighboring block and data
| 102 |         .unwrap_or(10_000); | support logic; impact depends on neighboring block and data flow. |
| 103 | (blank) | blank separator; no runtime behavior. |
| 104 |     if let Some(replay) = arg_value(&args, "--replay") { | runtime gate; condition changes can enable/disable c
| 105 |         let verify = has_flag(&args, "--verify"); | support logic; impact depends on neighboring block and data
| 106 |         println!("replay_file={} verify={} status=ok", replay, verify); | support logic; impact depends on neig
| 107 |         return; | support logic; impact depends on neighboring block and data flow. |
| 108 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 109 | (blank) | blank separator; no runtime behavior. |
| 110 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 111 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 112 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 113 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 114 | (blank) | blank separator; no runtime behavior. |
| 115 |     let mut lcg = seed; | support logic; impact depends on neighboring block and data flow. |
| 116 |     for i in 0..steps { | iteration logic; may alter timing, throughput, and safety constraints. |
| 117 |         // Simple deterministic pseudo-randomized demand profile by seed. | comment/documentation; changing aff
| 118 |         lcg = lcg.wrapping_mul(6364136223846793005).wrapping_add(1); | support logic; impact depends on neighbo
| 119 |         let n = ((lcg >> 33) as f64) / ((1u64 << 31) as f64); | support logic; impact depends on neighboring bl
| 120 |         sim.throttle_pct = (25.0 + n * 55.0).clamp(0.0, 100.0); | support logic; impact depends on neighboring
| 121 |         sim.brake_pct = if i % 400 > 360 { 20.0 } else { 0.0 }; | support logic; impact depends on neighboring
| 122 |         sim.loader_lift_cmd = if i % 140 < 70 { 0.5 } else { -0.4 }; | support logic; impact depends on neighbo
| 123 |         sim.loader_tilt_cmd = if i % 100 < 50 { 0.3 } else { -0.2 }; | support logic; impact depends on neighbo
| 124 |         sim.hitch_joystick = if i % 120 < 60 { 0.4 } else { -0.3 }; | support logic; impact depends on neighbor
| 125 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 126 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 127 | (blank) | blank separator; no runtime behavior. |
| 128 |     let summary = serde_json::json!({ | support logic; impact depends on neighboring block and data flow. |
| 129 |         "seed": seed, | support logic; impact depends on neighboring block and data flow. |
| 130 |         "model": model, | support logic; impact depends on neighboring block and data flow. |
| 131 |         "steps": steps, | support logic; impact depends on neighboring block and data flow. |
| 132 |         "elapsed_s": sim.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 133 |         "fuel_pct": sim.ecm.fuel_level_pct, | support logic; impact depends on neighboring block and data flow.
| 134 |         "oil_kpa": sim.ecm.oil_pressure_kpa, | support logic; impact depends on neighboring block and data flow
| 135 |         "can_health": sim.can_net.network_health_score_01(), | support logic; impact depends on neighboring bloc
| 136 |         "can_errors": sim.can_net.total_errors(), | support logic; impact depends on neighboring block and data
| 137 |         "loop_duration_ms": sim.metrics.loop_duration_ms, | support logic; impact depends on neighboring block
| 138 |         "steps_completed": sim.metrics.steps_completed, | support logic; impact depends on neighboring block an
| 139 |     }); | support logic; impact depends on neighboring block and data flow. |
| 140 | (blank) | blank separator; no runtime behavior. |
| 141 |     println!("{}", summary); | support logic; impact depends on neighboring block and data flow. |
```

| 142 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |

# File Deep Dive: src/boot_sequence.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 517 | Non-empty: 468 | Symbol count: 29

## Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| new | 12 |
| push_event | 9 |
| is_online | 3 |
| push | 3 |
| build_address_claim | 3 |
| fmt | 2 |
| update_interlocks | 2 |
| check_crank_inhibit | 2 |
| len | 2 |
| boot_ecu | 2 |
| check_all_critical_online | 2 |
| is_faulted | 1 |
| default | 1 |
| key_advance | 1 |
| key_off | 1 |
| tick | 1 |
| transmit | 1 |
| ecu_by_sa | 1 |
| build_id | 1 |
| from_raw | 1 |

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 19 | enum | IgnitionState | data/schema coupling and match/serde break risk |
| 32 | impl | std::fmt::Display for IgnitionState | method behavior and trait semantics |
| 33 | fn | fmt | API and behavior coupling to callers/implementers |
| 47 | enum | EcuBootStage | data/schema coupling and match/serde break risk |
| 66 | impl | std::fmt::Display for EcuBootStage | method behavior and trait semantics |
| 67 | fn | fmt | API and behavior coupling to callers/implementers |
| 85 | struct | EcuBootRecord | data/schema coupling and match/serde break risk |
| 98 | impl | EcuBootRecord | method behavior and trait semantics |
| 99 | fn | new | API and behavior coupling to callers/implementers |
| 111 | fn | is_online | API and behavior coupling to callers/implementers |
| 115 | fn | is_faulted | API and behavior coupling to callers/implementers |
| 123 | struct | BootEvent | data/schema coupling and match/serde break risk |
| 131 | enum | BootEventKind | data/schema coupling and match/serde break risk |
| 146 | struct | SafetyInterlock | data/schema coupling and match/serde break risk |
| 155 | struct | BootSequence | data/schema coupling and match/serde break risk |
| 177 | impl | Default for BootSequence | method behavior and trait semantics |
| 178 | fn | default | API and behavior coupling to callers/implementers |
| 183 | impl | BootSequence | method behavior and trait semantics |
| 184 | fn | new | API and behavior coupling to callers/implementers |
| 271 | fn | key_advance | API and behavior coupling to callers/implementers |
| 296 | fn | key_off | API and behavior coupling to callers/implementers |
| 313 | fn | tick | API and behavior coupling to callers/implementers |
| 360 | fn | boot_ecu | API and behavior coupling to callers/implementers |
| 453 | fn | update_interlocks | API and behavior coupling to callers/implementers |
| 461 | fn | check_crank_inhibit | API and behavior coupling to callers/implementers |
| 473 | fn | check_all_critical_online | API and behavior coupling to callers/implementers |
| 486 | fn | push_event | API and behavior coupling to callers/implementers |
| 500 | fn | ecu_by_sa | API and behavior coupling to callers/implementers |
| 508 | fn | build_address_claim | API and behavior coupling to callers/implementers |

## Line by Line Change Impact

| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Boot Sequence Module — models every step from turning the key to full operation. | comment/documentation; char
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |

```
| 3 | //! Implements the real power-up sequence of a heavy machine: | comment/documentation; changing affects understand
| 4 | //!   OFF → ACCESSORY → IGN_ON → SELF_TEST → ADDRESS_CLAIM → PRE_START_CHECKS | comment/documentation; changin
| 5 | //!   → CRANKING → START_CONFIRM → RUNNING | comment/documentation; changing affects understanding and intent tr
| 6 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 7 | //! J1939 Address Claiming (AC) procedure per SAE J1939-81: | comment/documentation; changing affects understandi
| 8 | //!   1. Node broadcasts its 64-bit NAME on null address (0xFE) | comment/documentation; changing affects understa
| 9 | //!   2. Waits 250 ms for conflicts | comment/documentation; changing affects understanding and intent traceabili
| 10 | //!   3. If no conflict: claims the desired SA | comment/documentation; changing affects understanding and intent
| 11 | //!   4. If conflict: lower NAME wins; loser tries next available SA or goes null | comment/documentation; changi
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | use crate::can_gateway::CanGateway; | import wiring; changing path/name can break compilation and module linkage
| 14 | use crate::j1939::{self, addr, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | // ■■ Ignition Key States ■■■
```

```
| 79 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 80 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 81 | (blank) | blank separator; no runtime behavior. |
| 82 | // ■■ Per-ECU boot record ■■■
```

```
| 155 | pub struct BootSequence { | data layout contract; field changes can break callers/serde/tests. |
| 156 |     pub ignition: IgnitionState, | support logic; impact depends on neighboring block and data flow. |
| 157 |     pub ecus: Vec<EcuBootRecord>, | support logic; impact depends on neighboring block and data flow. |
| 158 |     pub event_log: Vec<BootEvent>, | support logic; impact depends on neighboring block and data flow. |
| 159 |     pub safety_interlocks: Vec<SafetyInterlock>, | support logic; impact depends on neighboring block and data f
| 160 | (blank) | blank separator; no runtime behavior. |
| 161 |     // Global sequence timer (resets each state transition) | comment/documentation; changing affects understand
| 162 |     sequence_timer: f64, | support logic; impact depends on neighboring block and data flow. |
| 163 | (blank) | blank separator; no runtime behavior. |
| 164 |     // Are all critical ECUs online? | comment/documentation; changing affects understanding and intent traceab
| 165 |     pub all_critical_online: bool, | support logic; impact depends on neighboring block and data flow. |
| 166 | (blank) | blank separator; no runtime behavior. |
| 167 |     // Crank inhibit | comment/documentation; changing affects understanding and intent traceability. |
| 168 |     pub crank_inhibited: bool, | support logic; impact depends on neighboring block and data flow. |
| 169 |     pub crank_inhibit_reason: Option<&'static str>, | support logic; impact depends on neighboring block and da
| 170 | (blank) | blank separator; no runtime behavior. |
| 171 |     // Engine actually running (>400 RPM) — set by ECM | comment/documentation; changing affects understanding a
| 172 |     pub engine_running: bool, | support logic; impact depends on neighboring block and data flow. |
| 173 | (blank) | blank separator; no runtime behavior. |
| 174 |     pub elapsed: f64, | support logic; impact depends on neighboring block and data flow. |
| 175 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 176 | (blank) | blank separator; no runtime behavior. |
| 177 | impl Default for BootSequence { | implementation binding; behavior semantics and trait conformance live here. |
| 178 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 179 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 180 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 181 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 182 | (blank) | blank separator; no runtime behavior. |
| 183 | impl BootSequence { | implementation binding; behavior semantics and trait conformance live here. |
| 184 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 185 |         // Register all ECUs with their J1939 NAMEs. | comment/documentation; changing affects understanding an
| 186 |         // NAME bits: [62:60]=industry(0=global,2=ag), [59:56]=vehicle-system, | comment/documentation; changing
| 187 |         //            [55:52]=function, [47:40]=manufacturer, [28:21]=ECU-instance | comment/documentation; char
| 188 |         let ecus = vec![ | support logic; impact depends on neighboring block and data flow. |
| 189 |             EcuBootRecord::new("ECM #1", addr::ECM_1, 0x0000_6000_0000_0000), | support logic; impact depends o
| 190 |             EcuBootRecord::new("TCM", addr::TRANSMISSION, 0x0000_6000_0100_0000), | support logic; impact depen
| 191 |             EcuBootRecord::new("BCM", addr::CAB, 0x0000_6000_0200_0000), | support logic; impact depends on nei
| 192 |             EcuBootRecord::new("ICM/DASH", addr::INSTRUMENT, 0x0000_6000_0300_0000), | support logic; impact de
| 193 |             EcuBootRecord::new("HCM/HITCH", addr::HITCH, 0x0000_6000_0400_0000), | support logic; impact depend
| 194 |             EcuBootRecord::new("ABS/STAB", addr::BRAKES, 0x0000_6000_0500_0000), | support logic; impact depend
| 195 |             EcuBootRecord::new("ISOBUS-VT", addr::ISOBUS_VT, 0x0000_6000_0600_0000), | support logic; impact de
| 196 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 197 | (blank) | blank separator; no runtime behavior. |
| 198 |         // Real machine safety interlocks (prevents engine start if not met) | comment/documentation; changing a
| 199 |         let interlocks = vec![ | support logic; impact depends on neighboring block and data flow. |
| 200 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 201 |                 id: "TRANS_NEUTRAL", | support logic; impact depends on neighboring block and data flow. |
| 202 |                 description: "Transmission in neutral/park", | support logic; impact depends on neighboring bloc
| 203 |                 satisfied: false, | support logic; impact depends on neighboring block and data flow. |
| 204 |                 blocks_start: true, | support logic; impact depends on neighboring block and data flow. |
| 205 |             }, | support logic; impact depends on neighboring block and data flow. |
| 206 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 207 |                 id: "PARK_BRAKE", | support logic; impact depends on neighboring block and data flow. |
| 208 |                 description: "Parking brake applied", | support logic; impact depends on neighboring block and
| 209 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 210 |                 blocks_start: false, | support logic; impact depends on neighboring block and data flow. |
| 211 |             }, | support logic; impact depends on neighboring block and data flow. |
| 212 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 213 |                 id: "SEAT_OCCUPIED", | support logic; impact depends on neighboring block and data flow. |
| 214 |                 description: "Operator seat occupied", | support logic; impact depends on neighboring block and
| 215 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 216 |                 blocks_start: true, | support logic; impact depends on neighboring block and data flow. |
| 217 |             }, | support logic; impact depends on neighboring block and data flow. |
| 218 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 219 |                 id: "DOOR_CLOSED", | support logic; impact depends on neighboring block and data flow. |
| 220 |                 description: "Cab door closed", | support logic; impact depends on neighboring block and data f
| 221 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 222 |                 blocks_start: false, | support logic; impact depends on neighboring block and data flow. |
| 223 |             }, | support logic; impact depends on neighboring block and data flow. |
| 224 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 225 |                 id: "PTO_OFF", | support logic; impact depends on neighboring block and data flow. |
| 226 |                 description: "PTO disengaged", | support logic; impact depends on neighboring block and data flo
| 227 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 228 |                 blocks_start: false, | support logic; impact depends on neighboring block and data flow. |
| 229 |             }, | support logic; impact depends on neighboring block and data flow. |
| 230 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
```

```
| 231 |                 id: "HITCH_UP", | support logic; impact depends on neighboring block and data flow. |
| 232 |                 description: "Rear hitch raised for road", | support logic; impact depends on neighboring block
| 233 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 234 |                 blocks_start: false, | support logic; impact depends on neighboring block and data flow. |
| 235 |             }, | support logic; impact depends on neighboring block and data flow. |
| 236 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 237 |                 id: "OIL_PRESS_OK", | support logic; impact depends on neighboring block and data flow. |
| 238 |                 description: "Engine oil pressure in range", | support logic; impact depends on neighboring bloc
| 239 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 240 |                 blocks_start: false, | support logic; impact depends on neighboring block and data flow. |
| 241 |             }, | support logic; impact depends on neighboring block and data flow. |
| 242 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 243 |                 id: "COOLANT_OK", | support logic; impact depends on neighboring block and data flow. |
| 244 |                 description: "Coolant level OK", | support logic; impact depends on neighboring block and data
| 245 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 246 |                 blocks_start: false, | support logic; impact depends on neighboring block and data flow. |
| 247 |             }, | support logic; impact depends on neighboring block and data flow. |
| 248 |             SafetyInterlock { | support logic; impact depends on neighboring block and data flow. |
| 249 |                 id: "DEF_LEVEL_OK", | support logic; impact depends on neighboring block and data flow. |
| 250 |                 description: "DEF level ≥ 5%", | support logic; impact depends on neighboring block and data flo
| 251 |                 satisfied: true, | support logic; impact depends on neighboring block and data flow. |
| 252 |                 blocks_start: false, | support logic; impact depends on neighboring block and data flow. |
| 253 |             }, | support logic; impact depends on neighboring block and data flow. |
| 254 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 255 | (blank) | blank separator; no runtime behavior. |
| 256 |         BootSequence { | support logic; impact depends on neighboring block and data flow. |
| 257 |             ignition: IgnitionState::Off, | support logic; impact depends on neighboring block and data flow. |
| 258 |             ecus, | support logic; impact depends on neighboring block and data flow. |
| 259 |             event_log: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 260 |             safety_interlocks: interlocks, | support logic; impact depends on neighboring block and data flow.
| 261 |             sequence_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 262 |             all_critical_online: false, | support logic; impact depends on neighboring block and data flow. |
| 263 |             crank_inhibited: false, | support logic; impact depends on neighboring block and data flow. |
| 264 |             crank_inhibit_reason: None, | support logic; impact depends on neighboring block and data flow. |
| 265 |             engine_running: false, | support logic; impact depends on neighboring block and data flow. |
| 266 |             elapsed: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 267 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 268 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 269 | (blank) | blank separator; no runtime behavior. |
| 270 |     /// Advance the ignition key one step | comment/documentation; changing affects understanding and intent tra
| 271 |     pub fn key_advance(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 272 |         self.ignition = match self.ignition { | support logic; impact depends on neighboring block and data flow
| 273 |             IgnitionState::Off => IgnitionState::Accessory, | support logic; impact depends on neighboring bloc
| 274 |             IgnitionState::Accessory => IgnitionState::On, | support logic; impact depends on neighboring block
| 275 |             IgnitionState::On => { | support logic; impact depends on neighboring block and data flow. |
| 276 |                 if self.crank_inhibited { | runtime gate; condition changes can enable/disable critical paths.
| 277 |                     IgnitionState::On | support logic; impact depends on neighboring block and data flow. |
| 278 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 279 |                 // can't start | comment/documentation; changing affects understanding and intent traceability.
| 280 |                 else { | support logic; impact depends on neighboring block and data flow. |
| 281 |                     IgnitionState::Cranking | support logic; impact depends on neighboring block and data flow.
| 282 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 283 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 284 |             IgnitionState::Cranking => IgnitionState::On, // spring-return | support logic; impact depends on ne
| 285 |             IgnitionState::Running => IgnitionState::On,  // engine still running | support logic; impact depen
| 286 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 287 |         self.sequence_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 288 |         self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 289 |             "IGNITION", | support logic; impact depends on neighboring block and data flow. |
| 290 |             BootEventKind::PowerOn, | support logic; impact depends on neighboring block and data flow. |
| 291 |             format!("Key advanced to {:?}", self.ignition), | error propagation point; changes alter failure pro
| 292 |         ); | support logic; impact depends on neighboring block and data flow. |
| 293 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 294 | (blank) | blank separator; no runtime behavior. |
| 295 |     /// Turn key off | comment/documentation; changing affects understanding and intent traceability. |
| 296 |     pub fn key_off(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 297 |         self.ignition = IgnitionState::Off; | support logic; impact depends on neighboring block and data flow.
| 298 |         self.engine_running = false; | support logic; impact depends on neighboring block and data flow. |
| 299 |         // Power down all ECUs | comment/documentation; changing affects understanding and intent traceability.
| 300 |         for ecu in &mut self.ecus { | iteration logic; may alter timing, throughput, and safety constraints. |
| 301 |             ecu.stage = EcuBootStage::Unpowered; | support logic; impact depends on neighboring block and data
| 302 |             ecu.stage_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 303 |             ecu.online_at = None; | support logic; impact depends on neighboring block and data flow. |
| 304 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 305 |         self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 306 |             "IGNITION", | support logic; impact depends on neighboring block and data flow. |
```

```
| 307 |             BootEventKind::PowerOn, | support logic; impact depends on neighboring block and data flow. |
| 308 |             "Key OFF — all ECUs unpowered".into(), | support logic; impact depends on neighboring block and data
| 309 |         ); | support logic; impact depends on neighboring block and data flow. |
| 310 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 311 | (blank) | blank separator; no runtime behavior. |
| 312 |     /// Update the boot sequence state machine. Returns CAN frames to transmit. | comment/documentation; changin
| 313 |     pub fn tick( | function signature/body; changes affect API behavior and control-flow. |
| 314 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 315 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 316 |         gateway: &mut CanGateway, | support logic; impact depends on neighboring block and data flow. |
| 317 |         trans_neutral: bool, | support logic; impact depends on neighboring block and data flow. |
| 318 |         powered_sas: &[u8], | support logic; impact depends on neighboring block and data flow. |
| 319 |     ) -> Vec<J1939Frame> { | support logic; impact depends on neighboring block and data flow. |
| 320 |         self.elapsed += dt; | support logic; impact depends on neighboring block and data flow. |
| 321 |         self.sequence_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 322 | (blank) | blank separator; no runtime behavior. |
| 323 |         // Update safety interlocks | comment/documentation; changing affects understanding and intent traceabi
| 324 |         self.update_interlocks(trans_neutral); | support logic; impact depends on neighboring block and data fl
| 325 |         self.check_crank_inhibit(); | support logic; impact depends on neighboring block and data flow. |
| 326 | (blank) | blank separator; no runtime behavior. |
| 327 |         let mut frames: Vec<J1939Frame> = Vec::new(); | support logic; impact depends on neighboring block and
| 328 | (blank) | blank separator; no runtime behavior. |
| 329 |         for i in 0..self.ecus.len() { | iteration logic; may alter timing, throughput, and safety constraints.
| 330 |             let sa = self.ecus[i].sa; | support logic; impact depends on neighboring block and data flow. |
| 331 |             if powered_sas.contains(&sa) { | runtime gate; condition changes can enable/disable critical paths.
| 332 |                 self.boot_ecu(sa, dt, &mut frames); | support logic; impact depends on neighboring block and da
| 333 |             } else if self.ecus[i].stage != EcuBootStage::Unpowered { | support logic; impact depends on neighb
| 334 |                 self.ecus[i].stage = EcuBootStage::Unpowered; | support logic; impact depends on neighboring bl
| 335 |                 self.ecus[i].stage_timer = 0.0; | support logic; impact depends on neighboring block and data f
| 336 |                 self.ecus[i].online_at = None; | support logic; impact depends on neighboring block and data fl
| 337 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 338 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 339 | (blank) | blank separator; no runtime behavior. |
| 340 |         self.check_all_critical_online(); | support logic; impact depends on neighboring block and data flow. |
| 341 | (blank) | blank separator; no runtime behavior. |
| 342 |         // Crank → Running transition (engine fires) | comment/documentation; changing affects understanding a
| 343 |         if self.ignition == IgnitionState::Cranking && self.engine_running { | runtime gate; condition changes
| 344 |             self.ignition = IgnitionState::Running; | support logic; impact depends on neighboring block and da
| 345 |             self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 346 |                 "IGNITION", | support logic; impact depends on neighboring block and data flow. |
| 347 |                 BootEventKind::Running, | support logic; impact depends on neighboring block and data flow. |
| 348 |                 "Engine fired — transition to RUNNING".into(), | support logic; impact depends on neighboring b
| 349 |             ); | support logic; impact depends on neighboring block and data flow. |
| 350 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 351 | (blank) | blank separator; no runtime behavior. |
| 352 |         // Dispatch frames to gateway | comment/documentation; changing affects understanding and intent traceal
| 353 |         for f in &frames { | iteration logic; may alter timing, throughput, and safety constraints. |
| 354 |             gateway.transmit(f.clone()); | support logic; impact depends on neighboring block and data flow. |
| 355 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 356 |         frames | support logic; impact depends on neighboring block and data flow. |
| 357 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 358 | (blank) | blank separator; no runtime behavior. |
| 359 |     /// Run one ECU through its boot state machine | comment/documentation; changing affects understanding and
| 360 |     fn boot_ecu(&mut self, sa: u8, dt: f64, frames: &mut Vec<J1939Frame>) { | function signature/body; changes
| 361 |         let idx = match self.ecus.iter().position(\|e\| e.sa == sa) { | support logic; impact depends on neighb
| 362 |             Some(i) => i, | support logic; impact depends on neighboring block and data flow. |
| 363 |             None => return, | support logic; impact depends on neighboring block and data flow. |
| 364 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 365 |         let ts = self.elapsed; | support logic; impact depends on neighboring block and data flow. |
| 366 | (blank) | blank separator; no runtime behavior. |
| 367 |         // Copy all data we need BEFORE taking a mutable borrow of self | comment/documentation; changing affec
| 368 |         let stage = self.ecus[idx].stage; | support logic; impact depends on neighboring block and data flow. |
| 369 |         let stage_timer = self.ecus[idx].stage_timer; | support logic; impact depends on neighboring block and
| 370 |         let ecu_name = self.ecus[idx].name; | support logic; impact depends on neighboring block and data flow.
| 371 |         let ecu_sa = self.ecus[idx].sa; | support logic; impact depends on neighboring block and data flow. |
| 372 |         let j1939_name = self.ecus[idx].j1939_name; | support logic; impact depends on neighboring block and da
| 373 | (blank) | blank separator; no runtime behavior. |
| 374 |         self.ecus[idx].stage_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 375 | (blank) | blank separator; no runtime behavior. |
| 376 |         match stage { | branch dispatcher; arm changes alter behavior class handling. |
| 377 |             EcuBootStage::Unpowered => { | support logic; impact depends on neighboring block and data flow. |
| 378 |                 self.ecus[idx].stage = EcuBootStage::HardwareInit; | support logic; impact depends on neighbori
| 379 |                 self.ecus[idx].stage_timer = 0.0; | support logic; impact depends on neighboring block and data
| 380 |                 self.ecus[idx].boot_error = None; | support logic; impact depends on neighboring block and data
| 381 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 382 |             EcuBootStage::HardwareInit => { | support logic; impact depends on neighboring block and data flow.
```

```
| 383 |                 if stage_timer >= 0.050 { | runtime gate; condition changes can enable/disable critical paths.
| 384 |                     self.ecus[idx].stage = EcuBootStage::LoadingCalibration; | support logic; impact depends on
| 385 |                     self.ecus[idx].stage_timer = 0.0; | support logic; impact depends on neighboring block and
| 386 |                     self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 387 |                         ecu_name, | support logic; impact depends on neighboring block and data flow. |
| 388 |                         BootEventKind::StageChange, | support logic; impact depends on neighboring block and da
| 389 |                         format!("{}: HW init OK → loading calibration", ecu_name), | support logic; impact depe
| 390 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 391 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 392 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 393 |             EcuBootStage::LoadingCalibration => { | support logic; impact depends on neighboring block and data
| 394 |                 if stage_timer >= 0.080 { | runtime gate; condition changes can enable/disable critical paths.
| 395 |                     self.ecus[idx].stage = EcuBootStage::AddressClaiming; | support logic; impact depends on ne
| 396 |                     self.ecus[idx].stage_timer = 0.0; | support logic; impact depends on neighboring block and
| 397 |                     self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 398 |                         ecu_name, | support logic; impact depends on neighboring block and data flow. |
| 399 |                         BootEventKind::StageChange, | support logic; impact depends on neighboring block and da
| 400 |                         format!( | support logic; impact depends on neighboring block and data flow. |
| 401 |                             "{}: Calibration loaded → claiming SA 0x{:02X}", | support logic; impact depends on
| 402 |                             ecu_name, ecu_sa | support logic; impact depends on neighboring block and data flow
| 403 |                         ), | support logic; impact depends on neighboring block and data flow. |
| 404 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 405 |                     frames.push(build_address_claim(ts, j1939_name, ecu_sa)); | support logic; impact depends o
| 406 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 407 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 408 |             EcuBootStage::AddressClaiming => { | support logic; impact depends on neighboring block and data fl
| 409 |                 if stage_timer >= 0.250 { | runtime gate; condition changes can enable/disable critical paths.
| 410 |                     self.ecus[idx].stage = EcuBootStage::Handshaking; | support logic; impact depends on neighb
| 411 |                     self.ecus[idx].stage_timer = 0.0; | support logic; impact depends on neighboring block and
| 412 |                     self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 413 |                         ecu_name, | support logic; impact depends on neighboring block and data flow. |
| 414 |                         BootEventKind::AddressClaimed, | support logic; impact depends on neighboring block and
| 415 |                         format!( | support logic; impact depends on neighboring block and data flow. |
| 416 |                             "{}: Address 0x{:02X} claimed successfully", | support logic; impact depends on nei
| 417 |                             ecu_name, ecu_sa | support logic; impact depends on neighboring block and data flow
| 418 |                         ), | support logic; impact depends on neighboring block and data flow. |
| 419 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 420 |                     frames.push(build_address_claim(ts, j1939_name, ecu_sa)); | support logic; impact depends o
| 421 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 422 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 423 |             EcuBootStage::Handshaking => { | support logic; impact depends on neighboring block and data flow.
| 424 |                 if stage_timer >= 0.150 { | runtime gate; condition changes can enable/disable critical paths.
| 425 |                     self.ecus[idx].stage = EcuBootStage::PreOperational; | support logic; impact depends on nei
| 426 |                     self.ecus[idx].stage_timer = 0.0; | support logic; impact depends on neighboring block and
| 427 |                     self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 428 |                         ecu_name, | support logic; impact depends on neighboring block and data flow. |
| 429 |                         BootEventKind::HandshakeOk, | support logic; impact depends on neighboring block and da
| 430 |                         format!("{}: Peer handshake OK", ecu_name), | support logic; impact depends on neighbor
| 431 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 432 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 433 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 434 |             EcuBootStage::PreOperational => { | support logic; impact depends on neighboring block and data flo
| 435 |                 if stage_timer >= 0.200 { | runtime gate; condition changes can enable/disable critical paths.
| 436 |                     self.ecus[idx].stage = EcuBootStage::Running; | support logic; impact depends on neighborin
| 437 |                     self.ecus[idx].stage_timer = 0.0; | support logic; impact depends on neighboring block and
| 438 |                     self.ecus[idx].online_at = Some(ts); | support logic; impact depends on neighboring block a
| 439 |                     self.push_event( | support logic; impact depends on neighboring block and data flow. |
| 440 |                         ecu_name, | support logic; impact depends on neighboring block and data flow. |
| 441 |                         BootEventKind::Running, | support logic; impact depends on neighboring block and data f
| 442 |                         format!("{}: Pre-op checks passed → RUNNING at {:.3}s", ecu_name, ts), | support logic;
| 443 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 444 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 445 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 446 | (blank) | blank separator; no runtime behavior. |
| 447 |             EcuBootStage::Running \| EcuBootStage::Fault => { | support logic; impact depends on neighboring bl
| 448 |                 // Nothing more to do in boot state machine | comment/documentation; changing affects understan
| 449 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 450 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 451 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 452 | (blank) | blank separator; no runtime behavior. |
| 453 |     fn update_interlocks(&mut self, trans_neutral: bool) { | function signature/body; changes affect API behavi
| 454 |         for il in &mut self.safety_interlocks { | iteration logic; may alter timing, throughput, and safety con
| 455 |             if il.id == "TRANS_NEUTRAL" { | runtime gate; condition changes can enable/disable critical paths.
| 456 |                 il.satisfied = trans_neutral; | support logic; impact depends on neighboring block and data flo
| 457 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 458 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 459 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 460 | (blank) | blank separator; no runtime behavior. |
| 461 |     fn check_crank_inhibit(&mut self) { | function signature/body; changes affect API behavior and control-flow
| 462 |         self.crank_inhibited = false; | support logic; impact depends on neighboring block and data flow. |
| 463 |         self.crank_inhibit_reason = None; | support logic; impact depends on neighboring block and data flow. |
| 464 |         for il in &self.safety_interlocks { | iteration logic; may alter timing, throughput, and safety constra
| 465 |             if il.blocks_start && !il.satisfied { | runtime gate; condition changes can enable/disable critical
| 466 |                 self.crank_inhibited = true; | support logic; impact depends on neighboring block and data flow
| 467 |                 self.crank_inhibit_reason = Some(il.id); | support logic; impact depends on neighboring block a
| 468 |                 break; | support logic; impact depends on neighboring block and data flow. |
| 469 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 470 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 471 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 472 | (blank) | blank separator; no runtime behavior. |
| 473 |     fn check_all_critical_online(&mut self) { | function signature/body; changes affect API behavior and contro
| 474 |         // Critical ECUs: ECM and TCM | comment/documentation; changing affects understanding and intent traceal
| 475 |         let ecm_ok = self | support logic; impact depends on neighboring block and data flow. |
| 476 |             .ecus | support logic; impact depends on neighboring block and data flow. |
| 477 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 478 |             .any(\|e\| e.sa == addr::ECM_1 && e.is_online()); | support logic; impact depends on neighboring bl
| 479 |         let tcm_ok = self | support logic; impact depends on neighboring block and data flow. |
| 480 |             .ecus | support logic; impact depends on neighboring block and data flow. |
| 481 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 482 |             .any(\|e\| e.sa == addr::TRANSMISSION && e.is_online()); | support logic; impact depends on neighbo
| 483 |         self.all_critical_online = ecm_ok && tcm_ok; | support logic; impact depends on neighboring block and d
| 484 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 485 | (blank) | blank separator; no runtime behavior. |
| 486 |     fn push_event(&mut self, ecu: &'static str, kind: BootEventKind, desc: String) { | function signature/body;
| 487 |         self.event_log.push(BootEvent { | support logic; impact depends on neighboring block and data flow. |
| 488 |             timestamp: self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 489 |             ecu_name: ecu, | support logic; impact depends on neighboring block and data flow. |
| 490 |             event: kind, | support logic; impact depends on neighboring block and data flow. |
| 491 |             description: desc, | support logic; impact depends on neighboring block and data flow. |
| 492 |         }); | support logic; impact depends on neighboring block and data flow. |
| 493 |         // Keep only last 100 events | comment/documentation; changing affects understanding and intent traceab
| 494 |         if self.event_log.len() > 100 { | runtime gate; condition changes can enable/disable critical paths. |
| 495 |             self.event_log.remove(0); | support logic; impact depends on neighboring block and data flow. |
| 496 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 497 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 498 | (blank) | blank separator; no runtime behavior. |
| 499 |     /// Get ECU record by SA | comment/documentation; changing affects understanding and intent traceability. |
| 500 |     pub fn ecu_by_sa(&self, sa: u8) -> Option<&EcuBootRecord> { | function signature/body; changes affect API be
| 501 |         self.ecus.iter().find(\|e\| e.sa == sa) | support logic; impact depends on neighboring block and data f
| 502 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 503 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 504 | (blank) | blank separator; no runtime behavior. |
| 505 | // ■■ J1939 Address Claim frame builder ■■■
```

```
| update_with_measurement | 3 |
| forward_adas | 2 |
| surround_wide | 2 |
| predict | 2 |
| update | 1 |
| closest_object | 1 |
| fuse | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | enum | MarkType | data/schema coupling and match/serde break risk |
| 17 | impl | std::fmt::Display for MarkType | method behavior and trait semantics |
| 18 | fn | fmt | API and behavior coupling to callers/implementers |
| 33 | enum | SignType | data/schema coupling and match/serde break risk |
| 45 | impl | std::fmt::Display for SignType | method behavior and trait semantics |
| 46 | fn | fmt | API and behavior coupling to callers/implementers |
| 63 | struct | DetectedSign | data/schema coupling and match/serde break risk |
| 74 | enum | ObjectClass | data/schema coupling and match/serde break risk |
| 87 | impl | std::fmt::Display for ObjectClass | method behavior and trait semantics |
| 88 | fn | fmt | API and behavior coupling to callers/implementers |
| 105 | struct | DetectedObject | data/schema coupling and match/serde break risk |
| 122 | struct | LaneDetection | data/schema coupling and match/serde break risk |
| 141 | impl | Default for LaneDetection | method behavior and trait semantics |
| 142 | fn | default | API and behavior coupling to callers/implementers |
| 147 | impl | LaneDetection | method behavior and trait semantics |
| 148 | fn | new | API and behavior coupling to callers/implementers |
| 171 | struct | CameraSpec | data/schema coupling and match/serde break risk |
| 181 | impl | CameraSpec | method behavior and trait semantics |
| 182 | fn | forward_adas | API and behavior coupling to callers/implementers |
| 193 | fn | surround_wide | API and behavior coupling to callers/implementers |
| 207 | struct | CameraSystem | data/schema coupling and match/serde break risk |
| 245 | impl | Default for CameraSystem | method behavior and trait semantics |
| 246 | fn | default | API and behavior coupling to callers/implementers |
| 251 | impl | CameraSystem | method behavior and trait semantics |
| 252 | fn | new | API and behavior coupling to callers/implementers |
| 279 | fn | update | API and behavior coupling to callers/implementers |
| 407 | fn | closest_object | API and behavior coupling to callers/implementers |
| 419 | struct | FusedObject | data/schema coupling and match/serde break risk |
| 435 | impl | FusedObject | method behavior and trait semantics |
| 436 | fn | new | API and behavior coupling to callers/implementers |
| 458 | fn | predict | API and behavior coupling to callers/implementers |
| 467 | fn | update_with_measurement | API and behavior coupling to callers/implementers |
| 492 | struct | SensorFusion | data/schema coupling and match/serde break risk |
| 499 | impl | Default for SensorFusion | method behavior and trait semantics |
| 500 | fn | default | API and behavior coupling to callers/implementers |
| 505 | impl | SensorFusion | method behavior and trait semantics |
| 506 | fn | new | API and behavior coupling to callers/implementers |
| 515 | fn | fuse | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Camera-based perception — simulates automotive vision system. | comment/documentation; changing affects unders
| 2 | #![allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Models: forward camera (ADAS), surround-view, traffic sign recognition, | comment/documentation; changing affe
| 4 | //! object detection (YOLOv8-style outputs), lane detection with Hough transform. | comment/documentation; changin
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | // ■■ Lane mark type ■■■
```

```
| 21 |             MarkType::Dashed => "Dashed", | support logic; impact depends on neighboring block and data flow. |
| 22 |             MarkType::DoubleSolid => "Dbl-Solid", | support logic; impact depends on neighboring block and data f
| 23 |             MarkType::DottedShort => "Dotted", | support logic; impact depends on neighboring block and data flow
| 24 |             MarkType::GuardRail => "GuardRail", | support logic; impact depends on neighboring block and data flo
| 25 |             MarkType::None => "None", | support logic; impact depends on neighboring block and data flow. |
| 26 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 |         .fmt(f) | support logic; impact depends on neighboring block and data flow. |
| 28 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 29 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 30 | (blank) | blank separator; no runtime behavior. |
| 31 | // ■■ Traffic sign ■■■
```

```
| 97 |             ObjectClass::Barrier => write!(f, "Barrier   "), | protocol or I/O critical; changes may impact exte
| 98 |             ObjectClass::Animal => write!(f, "Animal    "), | protocol or I/O critical; changes may impact extern
| 99 |             ObjectClass::Unknown => write!(f, "Unknown   "), | protocol or I/O critical; changes may impact exte
| 100 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 101 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 102 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 103 | (blank) | blank separator; no runtime behavior. |
| 104 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 105 | pub struct DetectedObject { | data layout contract; field changes can break callers/serde/tests. |
| 106 |     pub id: u32, | support logic; impact depends on neighboring block and data flow. |
| 107 |     pub class: ObjectClass, | support logic; impact depends on neighboring block and data flow. |
| 108 |     pub confidence: f64, // 0-1 (YOLOv8-style score) | support logic; impact depends on neighboring block and da
| 109 |     pub distance_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 110 |     pub azimuth_deg: f64, // ±45° from forward | support logic; impact depends on neighboring block and data flo
| 111 |     pub width_m: f64,     // estimated physical width | support logic; impact depends on neighboring block and d
| 112 |     pub height_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 113 |     pub bbox_x: u16, // bounding box pixel coords | support logic; impact depends on neighboring block and data
| 114 |     pub bbox_y: u16, | support logic; impact depends on neighboring block and data flow. |
| 115 |     pub bbox_w: u16, | support logic; impact depends on neighboring block and data flow. |
| 116 |     pub bbox_h: u16, | support logic; impact depends on neighboring block and data flow. |
| 117 |     pub velocity_ms: Option<f64>, // optical flow estimate | support logic; impact depends on neighboring block
| 118 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 119 | (blank) | blank separator; no runtime behavior. |
| 120 | // ■■ Lane detection ■■■
```

```
| 173 |     pub resolution_h: u16, | support logic; impact depends on neighboring block and data flow. |
| 174 |     pub fov_h_deg: f64, // horizontal field of view | support logic; impact depends on neighboring block and dat
| 175 |     pub fov_v_deg: f64, | support logic; impact depends on neighboring block and data flow. |
| 176 |     pub fps: f64, | support logic; impact depends on neighboring block and data flow. |
| 177 |     pub focal_length_mm: f64, | support logic; impact depends on neighboring block and data flow. |
| 178 |     pub pixel_size_um: f64, | support logic; impact depends on neighboring block and data flow. |
| 179 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 180 | (blank) | blank separator; no runtime behavior. |
| 181 | impl CameraSpec { | implementation binding; behavior semantics and trait conformance live here. |
| 182 |     pub fn forward_adas() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 183 |         CameraSpec { | support logic; impact depends on neighboring block and data flow. |
| 184 |             resolution_w: 1920, | support logic; impact depends on neighboring block and data flow. |
| 185 |             resolution_h: 1080, | support logic; impact depends on neighboring block and data flow. |
| 186 |             fov_h_deg: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 187 |             fov_v_deg: 55.0, | support logic; impact depends on neighboring block and data flow. |
| 188 |             fps: 30.0, | support logic; impact depends on neighboring block and data flow. |
| 189 |             focal_length_mm: 7.7, | support logic; impact depends on neighboring block and data flow. |
| 190 |             pixel_size_um: 3.45, | support logic; impact depends on neighboring block and data flow. |
| 191 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 192 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 193 |     pub fn surround_wide() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 194 |         CameraSpec { | support logic; impact depends on neighboring block and data flow. |
| 195 |             resolution_w: 1280, | support logic; impact depends on neighboring block and data flow. |
| 196 |             resolution_h: 720, | support logic; impact depends on neighboring block and data flow. |
| 197 |             fov_h_deg: 190.0, | support logic; impact depends on neighboring block and data flow. |
| 198 |             fov_v_deg: 120.0, | support logic; impact depends on neighboring block and data flow. |
| 199 |             fps: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 200 |             focal_length_mm: 1.8, | support logic; impact depends on neighboring block and data flow. |
| 201 |             pixel_size_um: 4.0, | support logic; impact depends on neighboring block and data flow. |
| 202 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 203 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 204 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 205 | (blank) | blank separator; no runtime behavior. |
| 206 | // ■■ Vision system ■■■
```

```
| 249 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 250 | (blank) | blank separator; no runtime behavior. |
| 251 | impl CameraSystem { | implementation binding; behavior semantics and trait conformance live here. |
| 252 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 253 |         CameraSystem { | support logic; impact depends on neighboring block and data flow. |
| 254 |             forward_spec: CameraSpec::forward_adas(), | support logic; impact depends on neighboring block and d
| 255 |             rear_spec: CameraSpec::surround_wide(), | support logic; impact depends on neighboring block and dat
| 256 |             lane: LaneDetection::new(), | support logic; impact depends on neighboring block and data flow. |
| 257 |             objects_front: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 258 |             objects_rear: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 259 |             signs: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 260 |             active_speed_limit: Some(80), | support logic; impact depends on neighboring block and data flow. |
| 261 |             front_clear: true, | support logic; impact depends on neighboring block and data flow. |
| 262 |             rear_clear: true, | support logic; impact depends on neighboring block and data flow. |
| 263 |             left_clear: true, | support logic; impact depends on neighboring block and data flow. |
| 264 |             right_clear: true, | support logic; impact depends on neighboring block and data flow. |
| 265 |             inference_ms: 12.0, | support logic; impact depends on neighboring block and data flow. |
| 266 |             frames_processed: 0, | support logic; impact depends on neighboring block and data flow. |
| 267 |             detection_fps: 30.0, | support logic; impact depends on neighboring block and data flow. |
| 268 |             forward_cam_fault: false, | support logic; impact depends on neighboring block and data flow. |
| 269 |             rear_cam_fault: false, | support logic; impact depends on neighboring block and data flow. |
| 270 |             obscured: false, | support logic; impact depends on neighboring block and data flow. |
| 271 |             calibration_valid: true, | support logic; impact depends on neighboring block and data flow. |
| 272 |             update_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 273 |             noise_t: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 274 |             obj_id_counter: 0, | support logic; impact depends on neighboring block and data flow. |
| 275 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 276 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 277 | (blank) | blank separator; no runtime behavior. |
| 278 |     /// Update camera system — called each simulation tick. | comment/documentation; changing affects understand
| 279 |     pub fn update( | function signature/body; changes affect API behavior and control-flow. |
| 280 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 281 |         ego_speed_kmh: f64, | support logic; impact depends on neighboring block and data flow. |
| 282 |         _ego_heading: f64, | support logic; impact depends on neighboring block and data flow. |
| 283 |         radar_ttc: f64, | support logic; impact depends on neighboring block and data flow. |
| 284 |         elapsed: f64, | support logic; impact depends on neighboring block and data flow. |
| 285 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 286 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 287 |         self.noise_t += dt; | support logic; impact depends on neighboring block and data flow. |
| 288 |         self.update_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 289 | (blank) | blank separator; no runtime behavior. |
| 290 |         // Forward camera runs at 30 fps | comment/documentation; changing affects understanding and intent trac
| 291 |         if self.update_timer < 1.0 / self.forward_spec.fps { | runtime gate; condition changes can enable/disabl
| 292 |             return; | support logic; impact depends on neighboring block and data flow. |
| 293 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 294 |         self.update_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 295 |         self.frames_processed += 1; | support logic; impact depends on neighboring block and data flow. |
| 296 | (blank) | blank separator; no runtime behavior. |
| 297 |         let n = self.noise_t; | support logic; impact depends on neighboring block and data flow. |
| 298 |         let ns = \|s: f64\| ((s * 127.1 + 311.7).sin() * 43758.5).fract() - 0.5; | support logic; impact depends
| 299 | (blank) | blank separator; no runtime behavior. |
| 300 |         // ■ Lane detection (simulates Hough + polynomial fitting) ■■■■■■■■■■■■■■ | comment/documentation; ch
| 301 |         let speed_factor = (ego_speed_kmh / 60.0).clamp(0.0, 1.0); | support logic; impact depends on neighborin
| 302 |         self.lane.detected = ego_speed_kmh > 3.0 && !self.obscured; | support logic; impact depends on neighbori
| 303 |         self.lane.confidence = if self.lane.detected { | support logic; impact depends on neighboring block and
| 304 |             (0.92 + ns(n * 1.1) * 0.05).clamp(0.5, 0.99) | support logic; impact depends on neighboring block an
| 305 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 306 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 307 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 308 |         self.lane.ego_offset_m = ns(n * 0.3) * 0.15 * speed_factor; | support logic; impact depends on neighbori
| 309 |         self.lane.heading_error_deg = ns(n * 0.5) * 1.5 * speed_factor; | support logic; impact depends on neigh
| 310 |         self.lane.curvature_inv_m = ns(n * 0.01) * 0.002; | support logic; impact depends on neighboring block a
| 311 |         let lo = 1.75 + self.lane.ego_offset_m; | support logic; impact depends on neighboring block and data fl
| 312 |         let ro = 1.75 - self.lane.ego_offset_m; | support logic; impact depends on neighboring block and data fl
| 313 |         self.lane.left_offset_m = lo; | support logic; impact depends on neighboring block and data flow. |
| 314 |         self.lane.right_offset_m = ro; | support logic; impact depends on neighboring block and data flow. |
| 315 |         self.lane.lane_width_m = lo + ro; | support logic; impact depends on neighboring block and data flow. |
| 316 |         self.lane.departure_left = lo < 0.5; | support logic; impact depends on neighboring block and data flow
| 317 |         self.lane.departure_right = ro < 0.5; | support logic; impact depends on neighboring block and data flow
| 318 |         self.lane.ttld_s = if self.lane.departure_left \|\| self.lane.departure_right { | support logic; impact
| 319 |             lo.min(ro) / (ego_speed_kmh / 3.6).max(0.1) * 5.0 | support logic; impact depends on neighboring blo
| 320 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 321 |             f64::INFINITY | support logic; impact depends on neighboring block and data flow. |
| 322 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 323 | (blank) | blank separator; no runtime behavior. |
| 324 |         // ■ Object detection (simulated neural network outputs) ■■■■■■■■■■■■■■ | comment/documentation; char
```

```
| 325 |         self.objects_front.clear(); | support logic; impact depends on neighboring block and data flow. |
| 326 |         // Lead vehicle simulation | comment/documentation; changing affects understanding and intent traceabil
| 327 |         if radar_ttc < 30.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 328 |             let dist = (ego_speed_kmh / 3.6 * radar_ttc.min(10.0)).max(5.0); | support logic; impact depends on
| 329 |             self.objects_front.push(DetectedObject { | support logic; impact depends on neighboring block and da
| 330 |                 id: 1, | support logic; impact depends on neighboring block and data flow. |
| 331 |                 class: ObjectClass::Car, | support logic; impact depends on neighboring block and data flow. |
| 332 |                 confidence: 0.94 + ns(n * 2.1) * 0.04, | support logic; impact depends on neighboring block and
| 333 |                 distance_m: dist, | support logic; impact depends on neighboring block and data flow. |
| 334 |                 azimuth_deg: ns(n * 3.1) * 2.0, | support logic; impact depends on neighboring block and data f
| 335 |                 width_m: 1.9, | support logic; impact depends on neighboring block and data flow. |
| 336 |                 height_m: 1.5, | support logic; impact depends on neighboring block and data flow. |
| 337 |                 bbox_x: 760, | support logic; impact depends on neighboring block and data flow. |
| 338 |                 bbox_y: 400, | support logic; impact depends on neighboring block and data flow. |
| 339 |                 bbox_w: 120, | support logic; impact depends on neighboring block and data flow. |
| 340 |                 bbox_h: 80, | support logic; impact depends on neighboring block and data flow. |
| 341 |                 velocity_ms: Some((ego_speed_kmh - 5.0) / 3.6), | support logic; impact depends on neighboring
| 342 |             }); | support logic; impact depends on neighboring block and data flow. |
| 343 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 344 |         // Periodic additional objects | comment/documentation; changing affects understanding and intent tracea
| 345 |         let t_mod = elapsed % 20.0; | support logic; impact depends on neighboring block and data flow. |
| 346 |         if t_mod > 5.0 && t_mod < 15.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 347 |             self.objects_front.push(DetectedObject { | support logic; impact depends on neighboring block and da
| 348 |                 id: 2, | support logic; impact depends on neighboring block and data flow. |
| 349 |                 class: if (t_mod * 0.5).sin() > 0.0 { | support logic; impact depends on neighboring block and
| 350 |                     ObjectClass::Pedestrian | support logic; impact depends on neighboring block and data flow.
| 351 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 352 |                     ObjectClass::Cyclist | support logic; impact depends on neighboring block and data flow. |
| 353 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 354 |                 confidence: 0.72 + ns(n * 4.1) * 0.1, | support logic; impact depends on neighboring block and
| 355 |                 distance_m: 18.0 + ns(n * 5.1) * 5.0, | support logic; impact depends on neighboring block and
| 356 |                 azimuth_deg: 8.0 + ns(n * 6.1) * 3.0, | support logic; impact depends on neighboring block and
| 357 |                 width_m: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 358 |                 height_m: 1.8, | support logic; impact depends on neighboring block and data flow. |
| 359 |                 bbox_x: 1100, | support logic; impact depends on neighboring block and data flow. |
| 360 |                 bbox_y: 350, | support logic; impact depends on neighboring block and data flow. |
| 361 |                 bbox_w: 40, | support logic; impact depends on neighboring block and data flow. |
| 362 |                 bbox_h: 90, | support logic; impact depends on neighboring block and data flow. |
| 363 |                 velocity_ms: None, | support logic; impact depends on neighboring block and data flow. |
| 364 |             }); | support logic; impact depends on neighboring block and data flow. |
| 365 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 366 | (blank) | blank separator; no runtime behavior. |
| 367 |         // ■ Traffic sign recognition ■■■
```

```
| 401 | (blank) | blank separator; no runtime behavior. |
| 402 |         // ■ Camera health ■■■
```

```
| 477 |         self.distance_m = self.kf_x[0].max(0.0); | support logic; impact depends on neighboring block and data f
| 478 |         self.speed_ms = self.kf_x[2]; | support logic; impact depends on neighboring block and data flow. |
| 479 |         self.azimuth_deg = az; | support logic; impact depends on neighboring block and data flow. |
| 480 |         self.lateral_pos_m = az.to_radians().sin() * self.distance_m; | support logic; impact depends on neighbo
| 481 |         self.in_path = self.azimuth_deg.abs() < 5.0; | support logic; impact depends on neighboring block and da
| 482 |         self.ttc_s = if self.speed_ms > 0.5 { | support logic; impact depends on neighboring block and data flow
| 483 |             self.distance_m / self.speed_ms | support logic; impact depends on neighboring block and data flow.
| 484 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 485 |             f64::INFINITY | support logic; impact depends on neighboring block and data flow. |
| 486 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 487 |         self.sensor_sources \|= src_bit; | support logic; impact depends on neighboring block and data flow. |
| 488 |         self.confidence = (0.5 + self.sensor_sources.count_ones() as f64 * 0.15).min(0.99); | support logic; imp
| 489 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 490 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 491 | (blank) | blank separator; no runtime behavior. |
| 492 | pub struct SensorFusion { | data layout contract; field changes can break callers/serde/tests. |
| 493 |     pub objects: Vec<FusedObject>, | support logic; impact depends on neighboring block and data flow. |
| 494 |     pub ttc_critical: f64, // minimum TTC in path | support logic; impact depends on neighboring block and data
| 495 |     pub lead_dist_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 496 |     id_counter: u32, | support logic; impact depends on neighboring block and data flow. |
| 497 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 498 | (blank) | blank separator; no runtime behavior. |
| 499 | impl Default for SensorFusion { | implementation binding; behavior semantics and trait conformance live here. |
| 500 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 501 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 502 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 503 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 504 | (blank) | blank separator; no runtime behavior. |
| 505 | impl SensorFusion { | implementation binding; behavior semantics and trait conformance live here. |
| 506 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 507 |         SensorFusion { | support logic; impact depends on neighboring block and data flow. |
| 508 |             objects: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 509 |             ttc_critical: f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 510 |             lead_dist_m: f64::INFINITY, | support logic; impact depends on neighboring block and data flow. |
| 511 |             id_counter: 0, | support logic; impact depends on neighboring block and data flow. |
| 512 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 513 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 514 | (blank) | blank separator; no runtime behavior. |
| 515 |     pub fn fuse( | function signature/body; changes affect API behavior and control-flow. |
| 516 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 517 |         camera: &CameraSystem, | support logic; impact depends on neighboring block and data flow. |
| 518 |         radar_front_targets: &[crate::radar::RadarTarget], | support logic; impact depends on neighboring block
| 519 |         ego_speed_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 520 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 521 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 522 |         // Predict all existing tracks | comment/documentation; changing affects understanding and intent tracea
| 523 |         for obj in &mut self.objects { | iteration logic; may alter timing, throughput, and safety constraints.
| 524 |             obj.predict(dt); | support logic; impact depends on neighboring block and data flow. |
| 525 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 526 | (blank) | blank separator; no runtime behavior. |
| 527 |         // ■ Associate and update from RADAR (source bit 0) ■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentation,
| 528 |         for rt in radar_front_targets { | iteration logic; may alter timing, throughput, and safety constraints
| 529 |             let closing = if rt.range_rate_ms < 0.0 { | support logic; impact depends on neighboring block and
| 530 |                 -rt.range_rate_ms | support logic; impact depends on neighboring block and data flow. |
| 531 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 532 |                 0.0 | support logic; impact depends on neighboring block and data flow. |
| 533 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 534 |             if let Some(o) = self | runtime gate; condition changes can enable/disable critical paths. |
| 535 |                 .objects | support logic; impact depends on neighboring block and data flow. |
| 536 |                 .iter_mut() | support logic; impact depends on neighboring block and data flow. |
| 537 |                 .find(\|o\| (o.distance_m - rt.range_m).abs() < 8.0) | support logic; impact depends on neighbor
| 538 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 539 |                 o.update_with_measurement(rt.range_m, rt.azimuth_deg, closing, 0x01); | support logic; impact de
| 540 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 541 |                 let mut o = FusedObject::new( | support logic; impact depends on neighboring block and data flow
| 542 |                     self.id_counter, | support logic; impact depends on neighboring block and data flow. |
| 543 |                     ObjectClass::Car, | support logic; impact depends on neighboring block and data flow. |
| 544 |                     rt.range_m, | support logic; impact depends on neighboring block and data flow. |
| 545 |                     rt.azimuth_deg, | support logic; impact depends on neighboring block and data flow. |
| 546 |                     closing, | support logic; impact depends on neighboring block and data flow. |
| 547 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 548 |                 o.sensor_sources = 0x01; | support logic; impact depends on neighboring block and data flow. |
| 549 |                 self.objects.push(o); | support logic; impact depends on neighboring block and data flow. |
| 550 |                 self.id_counter += 1; | support logic; impact depends on neighboring block and data flow. |
| 551 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 552 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 553 | (blank) | blank separator; no runtime behavior. |
| 554 |         // ■ Associate and update from CAMERA (source bit 1) ■■■■■■■■■■■■■■■■■■■■ | comment/documentation;
| 555 |         for co in &camera.objects_front { | iteration logic; may alter timing, throughput, and safety constraint
| 556 |             if let Some(o) = self.objects.iter_mut().find(\|o\| { | runtime gate; condition changes can enable/d
| 557 |                 (o.distance_m - co.distance_m).abs() < 5.0 | support logic; impact depends on neighboring block
| 558 |                     && (o.azimuth_deg - co.azimuth_deg).abs() < 8.0 | support logic; impact depends on neighbori
| 559 |             }) { | support logic; impact depends on neighboring block and data flow. |
| 560 |                 let spd = co | support logic; impact depends on neighboring block and data flow. |
| 561 |                     .velocity_ms | support logic; impact depends on neighboring block and data flow. |
| 562 |                     .map(\|v\| (ego_speed_ms - v).max(0.0)) | support logic; impact depends on neighboring bloc
| 563 |                     .unwrap_or(o.speed_ms); | support logic; impact depends on neighboring block and data flow.
| 564 |                 o.update_with_measurement(co.distance_m, co.azimuth_deg, spd, 0x02); | support logic; impact dep
| 565 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 566 |                 let spd = co | support logic; impact depends on neighboring block and data flow. |
| 567 |                     .velocity_ms | support logic; impact depends on neighboring block and data flow. |
| 568 |                     .map(\|v\| (ego_speed_ms - v).max(0.0)) | support logic; impact depends on neighboring bloc
| 569 |                     .unwrap_or(5.0); | support logic; impact depends on neighboring block and data flow. |
| 570 |                 let mut o = FusedObject::new( | support logic; impact depends on neighboring block and data flo
| 571 |                     self.id_counter, | support logic; impact depends on neighboring block and data flow. |
| 572 |                     co.class, | support logic; impact depends on neighboring block and data flow. |
| 573 |                     co.distance_m, | support logic; impact depends on neighboring block and data flow. |
| 574 |                     co.azimuth_deg, | support logic; impact depends on neighboring block and data flow. |
| 575 |                     spd, | support logic; impact depends on neighboring block and data flow. |
| 576 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 577 |                 o.sensor_sources = 0x02; | support logic; impact depends on neighboring block and data flow. |
| 578 |                 self.objects.push(o); | support logic; impact depends on neighboring block and data flow. |
| 579 |                 self.id_counter += 1; | support logic; impact depends on neighboring block and data flow. |
| 580 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 581 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 582 | (blank) | blank separator; no runtime behavior. |
| 583 |         // Remove stale / out-of-range objects | comment/documentation; changing affects understanding and inter
| 584 |         self.objects | support logic; impact depends on neighboring block and data flow. |
| 585 |             .retain(\|o\| o.distance_m < 200.0 && o.confidence > 0.2); | support logic; impact depends on neigh
| 586 |         self.objects.truncate(64); | support logic; impact depends on neighboring block and data flow. |
| 587 | (blank) | blank separator; no runtime behavior. |
| 588 |         // Aggregate | comment/documentation; changing affects understanding and intent traceability. |
| 589 |         self.ttc_critical = self | support logic; impact depends on neighboring block and data flow. |
| 590 |             .objects | support logic; impact depends on neighboring block and data flow. |
| 591 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 592 |             .filter(\|o\| o.in_path && o.speed_ms > 0.1) | support logic; impact depends on neighboring block a
| 593 |             .map(\|o\| o.ttc_s) | support logic; impact depends on neighboring block and data flow. |
| 594 |             .fold(f64::INFINITY, f64::min); | support logic; impact depends on neighboring block and data flow.
| 595 |         self.lead_dist_m = self | support logic; impact depends on neighboring block and data flow. |
| 596 |             .objects | support logic; impact depends on neighboring block and data flow. |
| 597 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 598 |             .filter(\|o\| o.in_path) | support logic; impact depends on neighboring block and data flow. |
| 599 |             .map(\|o\| o.distance_m) | support logic; impact depends on neighboring block and data flow. |
| 600 |             .fold(f64::INFINITY, f64::min); | support logic; impact depends on neighboring block and data flow.
| 601 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 602 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/can_bus.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 180 | Non-empty: 173 | Symbol count: 8

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| publish | 17 |
| new | 2 |
| len | 1 |
| tick | 1 |
| broadcast_all | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 3 | const | BUS_LOAD_ALPHA | namespace and compile-time linkage |
| 6 | struct | CanFrame | data/schema coupling and match/serde break risk |
| 14 | struct | CanBus | data/schema coupling and match/serde break risk |
```

```
| 23 | impl | CanBus | method behavior and trait semantics |
| 24 | fn | new | API and behavior coupling to callers/implementers |
| 35 | fn | publish | API and behavior coupling to callers/implementers |
| 49 | fn | tick | API and behavior coupling to callers/implementers |
| 63 | fn | broadcast_all | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | const BUS_LOAD_ALPHA: f64 = 0.05; | support logic; impact depends on neighboring block and data flow. |
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 6 | pub struct CanFrame { | data layout contract; field changes can break callers/serde/tests. |
| 7 |     pub id: u32, | support logic; impact depends on neighboring block and data flow. |
| 8 |     pub name: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 9 |     pub value: String, | support logic; impact depends on neighboring block and data flow. |
| 10 |     pub timestamp: f64, | support logic; impact depends on neighboring block and data flow. |
| 11 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | /// Simulated 500 kbps CAN bus (HS-CAN) | comment/documentation; changing affects understanding and intent tracea
| 14 | pub struct CanBus { | data layout contract; field changes can break callers/serde/tests. |
| 15 |     pub frames: VecDeque<CanFrame>, | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub total_frames: u64, | support logic; impact depends on neighboring block and data flow. |
| 17 |     pub bus_load_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 18 |     pub frames_per_second: u32, | support logic; impact depends on neighboring block and data flow. |
| 19 |     frame_counter: u32, | support logic; impact depends on neighboring block and data flow. |
| 20 |     frame_timer: f64, | support logic; impact depends on neighboring block and data flow. |
| 21 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 22 | (blank) | blank separator; no runtime behavior. |
| 23 | impl CanBus { | implementation binding; behavior semantics and trait conformance live here. |
| 24 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 25 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 26 |             frames: VecDeque::new(), | support logic; impact depends on neighboring block and data flow. |
| 27 |             total_frames: 0, | support logic; impact depends on neighboring block and data flow. |
| 28 |             bus_load_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 29 |             frames_per_second: 0, | support logic; impact depends on neighboring block and data flow. |
| 30 |             frame_counter: 0, | support logic; impact depends on neighboring block and data flow. |
| 31 |             frame_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 32 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 |     pub fn publish(&mut self, id: u32, name: &'static str, value: String, ts: f64) { | function signature/body; c
| 36 |         self.frames.push_front(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 37 |             id, | support logic; impact depends on neighboring block and data flow. |
| 38 |             name, | support logic; impact depends on neighboring block and data flow. |
| 39 |             value, | support logic; impact depends on neighboring block and data flow. |
| 40 |             timestamp: ts, | support logic; impact depends on neighboring block and data flow. |
| 41 |         }); | support logic; impact depends on neighboring block and data flow. |
| 42 |         if self.frames.len() > 12 { | runtime gate; condition changes can enable/disable critical paths. |
| 43 |             self.frames.pop_back(); | support logic; impact depends on neighboring block and data flow. |
| 44 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 45 |         self.total_frames += 1; | support logic; impact depends on neighboring block and data flow. |
| 46 |         self.frame_counter += 1; | support logic; impact depends on neighboring block and data flow. |
| 47 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 | (blank) | blank separator; no runtime behavior. |
| 49 |     pub fn tick(&mut self, dt: f64) { | function signature/body; changes affect API behavior and control-flow. |
| 50 |         self.frame_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 51 |         if self.frame_timer >= 1.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 52 |             self.frames_per_second = self.frame_counter; | support logic; impact depends on neighboring block and
| 53 |             // 500kbps, each frame ~128 bits → ~3900 frames/s max | comment/documentation; changing affects unde
| 54 |             let load = self.frame_counter as f64 / 3900.0 * 100.0; | support logic; impact depends on neighboring
| 55 |             self.bus_load_pct += (load.min(100.0) - self.bus_load_pct) * BUS_LOAD_ALPHA; | support logic; impact
| 56 |             self.frame_counter = 0; | support logic; impact depends on neighboring block and data flow. |
| 57 |             self.frame_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 58 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 59 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 |     /// Publish all periodic ECU messages | comment/documentation; changing affects understanding and intent trac
| 62 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent trace
| 63 |     pub fn broadcast_all( | function signature/body; changes affect API behavior and control-flow. |
| 64 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 65 |         speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 66 |         rpm: f64, | support logic; impact depends on neighboring block and data flow. |
```

```
| 67 |         gear: &str, | support logic; impact depends on neighboring block and data flow. |
| 68 |         engine_temp: f64, | support logic; impact depends on neighboring block and data flow. |
| 69 |         throttle: f64, | support logic; impact depends on neighboring block and data flow. |
| 70 |         brake: f64, | support logic; impact depends on neighboring block and data flow. |
| 71 |         abs_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 72 |         esp_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 73 |         tcs_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 74 |         acc_enabled: bool, | support logic; impact depends on neighboring block and data flow. |
| 75 |         acc_speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 76 |         aeb_ttc: f64, | support logic; impact depends on neighboring block and data flow. |
| 77 |         lka_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 78 |         fuel_level: f64, | support logic; impact depends on neighboring block and data flow. |
| 79 |         boost: f64, | support logic; impact depends on neighboring block and data flow. |
| 80 |         torque: f64, | support logic; impact depends on neighboring block and data flow. |
| 81 |         oil_pressure: f64, | support logic; impact depends on neighboring block and data flow. |
| 82 |         elapsed: f64, | support logic; impact depends on neighboring block and data flow. |
| 83 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 84 |         self.publish(0x100, "VEH_SPEED", format!("{:06.2} km/h", speed), elapsed); | support logic; impact depend
| 85 |         self.publish(0x101, "ENGINE_RPM", format!("{:5.0} rpm", rpm), elapsed); | support logic; impact depends o
| 86 |         self.publish(0x102, "GEAR_POS", format!("{:<5}", gear), elapsed); | support logic; impact depends on neig
| 87 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 88 |             0x110, | support logic; impact depends on neighboring block and data flow. |
| 89 |             "ENG_TEMP", | support logic; impact depends on neighboring block and data flow. |
| 90 |             format!("{:5.1} °C", engine_temp), | support logic; impact depends on neighboring block and data flow
| 91 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 92 |         ); | support logic; impact depends on neighboring block and data flow. |
| 93 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 94 |             0x111, | support logic; impact depends on neighboring block and data flow. |
| 95 |             "THROTTLE", | support logic; impact depends on neighboring block and data flow. |
| 96 |             format!("{:5.1} %", throttle * 100.0), | support logic; impact depends on neighboring block and data
| 97 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 98 |         ); | support logic; impact depends on neighboring block and data flow. |
| 99 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 100 |             0x112, | support logic; impact depends on neighboring block and data flow. |
| 101 |             "BRAKE_REQ", | support logic; impact depends on neighboring block and data flow. |
| 102 |             format!("{:5.1} %", brake * 100.0), | support logic; impact depends on neighboring block and data f
| 103 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 104 |         ); | support logic; impact depends on neighboring block and data flow. |
| 105 |         self.publish(0x113, "TORQUE", format!("{:6.1} Nm", torque), elapsed); | support logic; impact depends o
| 106 |         self.publish(0x114, "BOOST", format!("{:4.2} bar", boost), elapsed); | support logic; impact depends on
| 107 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 108 |             0x115, | support logic; impact depends on neighboring block and data flow. |
| 109 |             "OIL_PRESS", | support logic; impact depends on neighboring block and data flow. |
| 110 |             format!("{:4.2} bar", oil_pressure), | support logic; impact depends on neighboring block and data
| 111 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 112 |         ); | support logic; impact depends on neighboring block and data flow. |
| 113 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 114 |             0x120, | support logic; impact depends on neighboring block and data flow. |
| 115 |             "FUEL_LEVEL", | support logic; impact depends on neighboring block and data flow. |
| 116 |             format!("{:5.1} %", fuel_level * 100.0), | support logic; impact depends on neighboring block and da
| 117 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 118 |         ); | support logic; impact depends on neighboring block and data flow. |
| 119 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 120 |             0x200, | support logic; impact depends on neighboring block and data flow. |
| 121 |             "ABS_STATUS", | support logic; impact depends on neighboring block and data flow. |
| 122 |             if abs_active { | runtime gate; condition changes can enable/disable critical paths. |
| 123 |                 "ACTIVE".into() | support logic; impact depends on neighboring block and data flow. |
| 124 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 125 |                 "IDLE".into() | support logic; impact depends on neighboring block and data flow. |
| 126 |             }, | support logic; impact depends on neighboring block and data flow. |
| 127 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 128 |         ); | support logic; impact depends on neighboring block and data flow. |
| 129 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 130 |             0x201, | support logic; impact depends on neighboring block and data flow. |
| 131 |             "ESP_STATUS", | support logic; impact depends on neighboring block and data flow. |
| 132 |             if esp_active { | runtime gate; condition changes can enable/disable critical paths. |
| 133 |                 "ACTIVE".into() | support logic; impact depends on neighboring block and data flow. |
| 134 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 135 |                 "IDLE".into() | support logic; impact depends on neighboring block and data flow. |
| 136 |             }, | support logic; impact depends on neighboring block and data flow. |
| 137 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 138 |         ); | support logic; impact depends on neighboring block and data flow. |
| 139 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 140 |             0x202, | support logic; impact depends on neighboring block and data flow. |
| 141 |             "TCS_STATUS", | support logic; impact depends on neighboring block and data flow. |
| 142 |             if tcs_active { | runtime gate; condition changes can enable/disable critical paths. |
```

```
| 143 |                 "ACTIVE".into() | support logic; impact depends on neighboring block and data flow. |
| 144 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 145 |                 "IDLE".into() | support logic; impact depends on neighboring block and data flow. |
| 146 |             }, | support logic; impact depends on neighboring block and data flow. |
| 147 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 148 |         ); | support logic; impact depends on neighboring block and data flow. |
| 149 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 150 |             0x300, | support logic; impact depends on neighboring block and data flow. |
| 151 |             "ACC_STATUS", | support logic; impact depends on neighboring block and data flow. |
| 152 |             if acc_enabled { | runtime gate; condition changes can enable/disable critical paths. |
| 153 |                 format!("{:3.0}km/h", acc_speed) | support logic; impact depends on neighboring block and data f
| 154 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 155 |                 "OFF  ".into() | support logic; impact depends on neighboring block and data flow. |
| 156 |             }, | support logic; impact depends on neighboring block and data flow. |
| 157 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 158 |         ); | support logic; impact depends on neighboring block and data flow. |
| 159 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 160 |             0x301, | support logic; impact depends on neighboring block and data flow. |
| 161 |             "AEB_TTC", | support logic; impact depends on neighboring block and data flow. |
| 162 |             if aeb_ttc.is_infinite() { | runtime gate; condition changes can enable/disable critical paths. |
| 163 |                 " ∞ s".into() | support logic; impact depends on neighboring block and data flow. |
| 164 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 165 |                 format!("{:4.1} s", aeb_ttc) | support logic; impact depends on neighboring block and data flow
| 166 |             }, | support logic; impact depends on neighboring block and data flow. |
| 167 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 168 |         ); | support logic; impact depends on neighboring block and data flow. |
| 169 |         self.publish( | support logic; impact depends on neighboring block and data flow. |
| 170 |             0x302, | support logic; impact depends on neighboring block and data flow. |
| 171 |             "LKA_STATUS", | support logic; impact depends on neighboring block and data flow. |
| 172 |             if lka_active { | runtime gate; condition changes can enable/disable critical paths. |
| 173 |                 "ACTIVE".into() | support logic; impact depends on neighboring block and data flow. |
| 174 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 175 |                 "IDLE".into() | support logic; impact depends on neighboring block and data flow. |
| 176 |             }, | support logic; impact depends on neighboring block and data flow. |
| 177 |             elapsed, | support logic; impact depends on neighboring block and data flow. |
| 178 |         ); | support logic; impact depends on neighboring block and data flow. |
| 179 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 180 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/can_gateway.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 317 | Non-empty: 280 | Symbol count: 26

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 9 |
| update_state | 4 |
| push | 4 |
| log | 3 |
| len | 3 |
| record_tx_error | 2 |
| record_success | 2 |
| tick | 2 |
| log_error | 2 |
| fmt | 1 |
| record_rx_error | 1 |
| default | 1 |
| register_node | 1 |
| set_node_online | 1 |
| transmit | 1 |
| process | 1 |
| bits | 1 |
| receive_for | 1 |
| analyzer_frames | 1 |
| kill_node | 1 |
| revive_node | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
```

```
|---:|---|---|---|
| 13 | enum | BusState | data/schema coupling and match/serde break risk |
| 22 | impl | std::fmt::Display for BusState | method behavior and trait semantics |
| 23 | fn | fmt | API and behavior coupling to callers/implementers |
| 34 | struct | BusError | data/schema coupling and match/serde break risk |
| 42 | enum | BusErrorKind | data/schema coupling and match/serde break risk |
| 54 | struct | CanNode | data/schema coupling and match/serde break risk |
| 64 | impl | CanNode | method behavior and trait semantics |
| 65 | fn | new | API and behavior coupling to callers/implementers |
| 78 | fn | record_tx_error | API and behavior coupling to callers/implementers |
| 83 | fn | record_rx_error | API and behavior coupling to callers/implementers |
| 88 | fn | record_success | API and behavior coupling to callers/implementers |
| 95 | fn | update_state | API and behavior coupling to callers/implementers |
| 109 | struct | CanGateway | data/schema coupling and match/serde break risk |
| 142 | impl | Default for CanGateway | method behavior and trait semantics |
| 143 | fn | default | API and behavior coupling to callers/implementers |
| 148 | impl | CanGateway | method behavior and trait semantics |
| 149 | fn | new | API and behavior coupling to callers/implementers |
| 170 | fn | register_node | API and behavior coupling to callers/implementers |
| 177 | fn | set_node_online | API and behavior coupling to callers/implementers |
| 186 | fn | transmit | API and behavior coupling to callers/implementers |
| 192 | fn | tick | API and behavior coupling to callers/implementers |
| 275 | fn | receive_for | API and behavior coupling to callers/implementers |
| 282 | fn | analyzer_frames | API and behavior coupling to callers/implementers |
| 286 | fn | log_error | API and behavior coupling to callers/implementers |
| 300 | fn | kill_node | API and behavior coupling to callers/implementers |
| 309 | fn | revive_node | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Central CAN Bus Gateway — routes frames between all ECUs, monitors bus health. | comment/documentation; changi
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Models a real HS-CAN (500 kbps) bus as used in heavy machinery. | comment/documentation; changing affects unde
| 4 | //! Implements J1939 network management: address claiming, bus-off recovery, | comment/documentation; changing af
| 5 | //! error counting per ISO 11898-1 (TEC/REC counters), and frame arbitration. | comment/documentation; changing a
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | use crate::j1939::{J1939Bus, J1939Frame}; | import wiring; changing path/name can break compilation and module li
| 8 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | // ■■ Bus State Machine (ISO 11898-1) ■■■
```

```
| 45 |     CrcError, | support logic; impact depends on neighboring block and data flow. |
| 46 |     FormError, | support logic; impact depends on neighboring block and data flow. |
| 47 |     AcknowledgeError, | support logic; impact depends on neighboring block and data flow. |
| 48 |     Overload, | support logic; impact depends on neighboring block and data flow. |
| 49 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 50 | (blank) | blank separator; no runtime behavior. |
| 51 | // ■■ Registered Node ■■■
```

```
| 121 | (blank) | blank separator; no runtime behavior. |
| 122 |     /// Bus-level statistics | comment/documentation; changing affects understanding and intent traceability. |
| 123 |     pub total_tx: u64, | support logic; impact depends on neighboring block and data flow. |
| 124 |     pub total_rx: u64, | support logic; impact depends on neighboring block and data flow. |
| 125 |     pub total_errors: u64, | support logic; impact depends on neighboring block and data flow. |
| 126 |     pub bus_state: BusState, | support logic; impact depends on neighboring block and data flow. |
| 127 | (blank) | blank separator; no runtime behavior. |
| 128 |     /// Error log (last 64) | comment/documentation; changing affects understanding and intent traceability. |
| 129 |     pub error_log: VecDeque<BusError>, | support logic; impact depends on neighboring block and data flow. |
| 130 | (blank) | blank separator; no runtime behavior. |
| 131 |     /// Bus load (updated every second) | comment/documentation; changing affects understanding and intent trace
| 132 |     pub bus_load_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 133 |     frame_counter_1s: u32, | support logic; impact depends on neighboring block and data flow. |
| 134 |     time_accumulator: f64, | support logic; impact depends on neighboring block and data flow. |
| 135 | (blank) | blank separator; no runtime behavior. |
| 136 |     // Fault injection flags (for the fault panel) | comment/documentation; changing affects understanding and i
| 137 |     pub inject_bit_error: bool, | support logic; impact depends on neighboring block and data flow. |
| 138 |     pub inject_bus_off: bool, | support logic; impact depends on neighboring block and data flow. |
| 139 |     pub inject_missing_ack: bool, | support logic; impact depends on neighboring block and data flow. |
| 140 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 141 | (blank) | blank separator; no runtime behavior. |
| 142 | impl Default for CanGateway { | implementation binding; behavior semantics and trait conformance live here. |
| 143 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 144 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 145 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 146 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 147 | (blank) | blank separator; no runtime behavior. |
| 148 | impl CanGateway { | implementation binding; behavior semantics and trait conformance live here. |
| 149 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 150 |         CanGateway { | support logic; impact depends on neighboring block and data flow. |
| 151 |             bus: J1939Bus::new(), | support logic; impact depends on neighboring block and data flow. |
| 152 |             tx_queue: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 153 |             dispatched: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 154 |             nodes: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 155 |             total_tx: 0, | support logic; impact depends on neighboring block and data flow. |
| 156 |             total_rx: 0, | support logic; impact depends on neighboring block and data flow. |
| 157 |             total_errors: 0, | support logic; impact depends on neighboring block and data flow. |
| 158 |             bus_state: BusState::ErrorActive, | support logic; impact depends on neighboring block and data flow
| 159 |             error_log: VecDeque::new(), | support logic; impact depends on neighboring block and data flow. |
| 160 |             bus_load_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 161 |             frame_counter_1s: 0, | support logic; impact depends on neighboring block and data flow. |
| 162 |             time_accumulator: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 163 |             inject_bit_error: false, | support logic; impact depends on neighboring block and data flow. |
| 164 |             inject_bus_off: false, | support logic; impact depends on neighboring block and data flow. |
| 165 |             inject_missing_ack: false, | support logic; impact depends on neighboring block and data flow. |
| 166 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 167 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 168 | (blank) | blank separator; no runtime behavior. |
| 169 |     /// Register an ECU on the bus | comment/documentation; changing affects understanding and intent traceabili
| 170 |     pub fn register_node(&mut self, sa: u8, name: &'static str) { | function signature/body; changes affect API
| 171 |         if !self.nodes.iter().any(\|n\| n.source_addr == sa) { | runtime gate; condition changes can enable/disa
| 172 |             self.nodes.push(CanNode::new(sa, name)); | support logic; impact depends on neighboring block and da
| 173 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 174 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 175 | (blank) | blank separator; no runtime behavior. |
| 176 |     /// Mark a node as online (called after successful address claim) | comment/documentation; changing affects
| 177 |     pub fn set_node_online(&mut self, sa: u8) { | function signature/body; changes affect API behavior and cont
| 178 |         if let Some(node) = self.nodes.iter_mut().find(\|n\| n.source_addr == sa) { | runtime gate; condition ch
| 179 |             node.online = true; | support logic; impact depends on neighboring block and data flow. |
| 180 |             node.tec = 0; | support logic; impact depends on neighboring block and data flow. |
| 181 |             node.rec = 0; | support logic; impact depends on neighboring block and data flow. |
| 182 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 183 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 184 | (blank) | blank separator; no runtime behavior. |
| 185 |     /// Called by each ECU to submit a frame for transmission | comment/documentation; changing affects understa
| 186 |     pub fn transmit(&mut self, frame: J1939Frame) { | function signature/body; changes affect API behavior and c
| 187 |         self.tx_queue.push(frame); | support logic; impact depends on neighboring block and data flow. |
| 188 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 189 | (blank) | blank separator; no runtime behavior. |
| 190 |     /// Tick: arbitrate all queued frames, dispatch them, update counters. | comment/documentation; changing af
| 191 |     /// Call once per simulation step. | comment/documentation; changing affects understanding and intent tracea
| 192 |     pub fn tick(&mut self, dt: f64) { | function signature/body; changes affect API behavior and control-flow.
| 193 |         // ■■ Fault injection ■■■
```

```
| 197 |             self.log_error( | support logic; impact depends on neighboring block and data flow. |
| 198 |                 0.0, | support logic; impact depends on neighboring block and data flow. |
| 199 |                 BusErrorKind::BitError, | support logic; impact depends on neighboring block and data flow. |
| 200 |                 None, | support logic; impact depends on neighboring block and data flow. |
| 201 |                 "INJECTED: Bus-Off condition", | support logic; impact depends on neighboring block and data flo
| 202 |             ); | support logic; impact depends on neighboring block and data flow. |
| 203 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 204 | (blank) | blank separator; no runtime behavior. |
| 205 |         // ■■ Arbitration: sort by CAN ID (lower ID = higher priority) ■■■■■■■■ | comment/documentation; char
| 206 |         self.tx_queue.sort_by_key(\|f\| f.raw_id); | support logic; impact depends on neighboring block and data
| 207 | (blank) | blank separator; no runtime behavior. |
| 208 |         self.dispatched.clear(); | support logic; impact depends on neighboring block and data flow. |
| 209 | (blank) | blank separator; no runtime behavior. |
| 210 |         // Collect frames to process (avoids borrow conflict with self.log_error) | comment/documentation; chang
| 211 |         let frames: Vec<J1939Frame> = self.tx_queue.drain(..).collect(); | support logic; impact depends on neig
| 212 | (blank) | blank separator; no runtime behavior. |
| 213 |         for frame in frames { | iteration logic; may alter timing, throughput, and safety constraints. |
| 214 |             // Fault injection: drop frame + log error | comment/documentation; changing affects understanding a
| 215 |             if self.inject_bit_error && self.total_tx.is_multiple_of(100) { | runtime gate; condition changes ca
| 216 |                 self.total_errors += 1; | support logic; impact depends on neighboring block and data flow. |
| 217 |                 let ts = frame.timestamp; | support logic; impact depends on neighboring block and data flow. |
| 218 |                 let sa = frame.sa; | support logic; impact depends on neighboring block and data flow. |
| 219 |                 self.error_log.push_front(BusError { | support logic; impact depends on neighboring block and da
| 220 |                     timestamp: ts, | support logic; impact depends on neighboring block and data flow. |
| 221 |                     kind: BusErrorKind::BitError, | support logic; impact depends on neighboring block and data
| 222 |                     source_sa: Some(sa), | support logic; impact depends on neighboring block and data flow. |
| 223 |                     description: "INJECTED: Bit error", | support logic; impact depends on neighboring block and
| 224 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 225 |                 if self.error_log.len() > 64 { | runtime gate; condition changes can enable/disable critical pa
| 226 |                     self.error_log.pop_back(); | support logic; impact depends on neighboring block and data flo
| 227 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 228 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 229 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 230 |             if self.inject_missing_ack && self.total_tx.is_multiple_of(50) { | runtime gate; condition changes c
| 231 |                 self.total_errors += 1; | support logic; impact depends on neighboring block and data flow. |
| 232 |                 let ts = frame.timestamp; | support logic; impact depends on neighboring block and data flow. |
| 233 |                 let sa = frame.sa; | support logic; impact depends on neighboring block and data flow. |
| 234 |                 self.error_log.push_front(BusError { | support logic; impact depends on neighboring block and da
| 235 |                     timestamp: ts, | support logic; impact depends on neighboring block and data flow. |
| 236 |                     kind: BusErrorKind::AcknowledgeError, | support logic; impact depends on neighboring block a
| 237 |                     source_sa: Some(sa), | support logic; impact depends on neighboring block and data flow. |
| 238 |                     description: "INJECTED: ACK missing", | support logic; impact depends on neighboring block a
| 239 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 240 |                 if self.error_log.len() > 64 { | runtime gate; condition changes can enable/disable critical pa
| 241 |                     self.error_log.pop_back(); | support logic; impact depends on neighboring block and data flo
| 242 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 243 |                 if let Some(node) = self.nodes.iter_mut().find(\|n\| n.source_addr == sa) { | runtime gate; cond
| 244 |                     node.record_tx_error(); | support logic; impact depends on neighboring block and data flow.
| 245 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 246 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 247 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 248 | (blank) | blank separator; no runtime behavior. |
| 249 |             // Update node stats | comment/documentation; changing affects understanding and intent traceability
| 250 |             if let Some(node) = self.nodes.iter_mut().find(\|n\| n.source_addr == frame.sa) { | runtime gate; co
| 251 |                 node.last_tx_ts = frame.timestamp; | support logic; impact depends on neighboring block and data
| 252 |                 node.record_success(); | support logic; impact depends on neighboring block and data flow. |
| 253 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 254 |             self.bus.push(frame.clone()); | support logic; impact depends on neighboring block and data flow. |
| 255 |             self.dispatched.push(frame); | support logic; impact depends on neighboring block and data flow. |
| 256 |             self.total_tx += 1; | support logic; impact depends on neighboring block and data flow. |
| 257 |             self.frame_counter_1s += 1; | support logic; impact depends on neighboring block and data flow. |
| 258 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 259 | (blank) | blank separator; no runtime behavior. |
| 260 |         // ■■ Bus load calculation ■■■■■■■■
```

```
| 273 |     /// Get all frames dispatched this cycle visible to a given SA. | comment/documentation; changing affects u
| 274 |     /// Returns: broadcast frames + peer-to-peer frames addressed to `sa`. | comment/documentation; changing af
| 275 |     pub fn receive_for(&self, sa: u8) -> impl Iterator<Item = &J1939Frame> { | function signature/body; changes
| 276 |         self.dispatched | support logic; impact depends on neighboring block and data flow. |
| 277 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 278 |             .filter(move \|f\| f.sa != sa && (f.da == 0xFF \|\| f.da == sa)) | support logic; impact depends on
| 279 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 280 | (blank) | blank separator; no runtime behavior. |
| 281 |     /// Peek at latest N frames from the bus log (for CAN Analyzer) | comment/documentation; changing affects u
| 282 |     pub fn analyzer_frames(&self, count: usize) -> impl Iterator<Item = &J1939Frame> { | function signature/bod
| 283 |         self.bus.frames.iter().take(count) | support logic; impact depends on neighboring block and data flow.
| 284 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 285 | (blank) | blank separator; no runtime behavior. |
| 286 |     fn log_error(&mut self, ts: f64, kind: BusErrorKind, sa: Option<u8>, desc: &'static str) { | function signat
| 287 |         self.error_log.push_front(BusError { | support logic; impact depends on neighboring block and data flow
| 288 |             timestamp: ts, | support logic; impact depends on neighboring block and data flow. |
| 289 |             kind, | support logic; impact depends on neighboring block and data flow. |
| 290 |             source_sa: sa, | support logic; impact depends on neighboring block and data flow. |
| 291 |             description: desc, | support logic; impact depends on neighboring block and data flow. |
| 292 |         }); | support logic; impact depends on neighboring block and data flow. |
| 293 |         if self.error_log.len() > 64 { | runtime gate; condition changes can enable/disable critical paths. |
| 294 |             self.error_log.pop_back(); | support logic; impact depends on neighboring block and data flow. |
| 295 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 296 |         self.total_errors += 1; | support logic; impact depends on neighboring block and data flow. |
| 297 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 298 | (blank) | blank separator; no runtime behavior. |
| 299 |     /// Simulate a node going offline (e.g., ECU power cut) | comment/documentation; changing affects understan
| 300 |     pub fn kill_node(&mut self, sa: u8) { | function signature/body; changes affect API behavior and control-fl
| 301 |         if let Some(node) = self.nodes.iter_mut().find(\|n\| n.source_addr == sa) { | runtime gate; condition c
| 302 |             node.online = false; | support logic; impact depends on neighboring block and data flow. |
| 303 |             node.tec = 256; | support logic; impact depends on neighboring block and data flow. |
| 304 |             node.state = BusState::BusOff; | support logic; impact depends on neighboring block and data flow.
| 305 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 306 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 307 | (blank) | blank separator; no runtime behavior. |
| 308 |     /// Revive a killed node | comment/documentation; changing affects understanding and intent traceability. |
| 309 |     pub fn revive_node(&mut self, sa: u8) { | function signature/body; changes affect API behavior and control-
| 310 |         if let Some(node) = self.nodes.iter_mut().find(\|n\| n.source_addr == sa) { | runtime gate; condition c
| 311 |             node.online = true; | support logic; impact depends on neighboring block and data flow. |
| 312 |             node.tec = 0; | support logic; impact depends on neighboring block and data flow. |
| 313 |             node.rec = 0; | support logic; impact depends on neighboring block and data flow. |
| 314 |             node.state = BusState::ErrorActive; | support logic; impact depends on neighboring block and data f
| 315 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 316 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 317 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/can_network.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 1381 | Non-empty: 1292 | Symbol count: 73

### Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| new | 63 |
| len | 14 |
| tick | 12 |
| register | 11 |
| transmit | 9 |
| push | 8 |
| bus_mut | 7 |
| from_raw | 6 |
| build_id | 6 |
| log_error | 6 |
| update_state | 4 |
| tx_error | 4 |
| bus_ref | 4 |
| kbps | 3 |
| fmt | 3 |
| set_online | 3 |
| online_count | 3 |
| snapshot | 3 |

```
| max_fps | 2 |
| tx_success | 2 |
| default | 2 |
| send_bam | 2 |
| process_rx | 2 |
| should_fire | 2 |
| build_message_matrix | 2 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 37 | enum | CanSpeed | data/schema coupling and match/serde break risk |
| 48 | impl | CanSpeed | method behavior and trait semantics |
| 49 | fn | kbps | API and behavior coupling to callers/implementers |
| 53 | fn | max_fps | API and behavior coupling to callers/implementers |
| 68 | enum | BusId | data/schema coupling and match/serde break risk |
| 77 | impl | std::fmt::Display for BusId | method behavior and trait semantics |
| 78 | fn | fmt | API and behavior coupling to callers/implementers |
| 92 | enum | BusState | data/schema coupling and match/serde break risk |
| 98 | impl | std::fmt::Display for BusState | method behavior and trait semantics |
| 99 | fn | fmt | API and behavior coupling to callers/implementers |
| 110 | enum | CanErrorKind | data/schema coupling and match/serde break risk |
| 120 | impl | std::fmt::Display for CanErrorKind | method behavior and trait semantics |
| 121 | fn | fmt | API and behavior coupling to callers/implementers |
| 136 | struct | BusError | data/schema coupling and match/serde break risk |
| 147 | struct | CanNode | data/schema coupling and match/serde break risk |
| 166 | impl | CanNode | method behavior and trait semantics |
| 167 | fn | new | API and behavior coupling to callers/implementers |
| 185 | fn | tx_success | API and behavior coupling to callers/implementers |
| 189 | fn | tx_error | API and behavior coupling to callers/implementers |
| 193 | fn | rx_error | API and behavior coupling to callers/implementers |
| 198 | fn | update_state | API and behavior coupling to callers/implementers |
| 215 | struct | TpSession | data/schema coupling and match/serde break risk |
| 228 | struct | CanTransportProtocol | data/schema coupling and match/serde break risk |
| 237 | impl | Default for CanTransportProtocol | method behavior and trait semantics |
| 238 | fn | default | API and behavior coupling to callers/implementers |
| 243 | impl | CanTransportProtocol | method behavior and trait semantics |
| 244 | fn | new | API and behavior coupling to callers/implementers |
| 253 | fn | send_bam | API and behavior coupling to callers/implementers |
| 293 | fn | process_rx | API and behavior coupling to callers/implementers |
| 346 | fn | tick | API and behavior coupling to callers/implementers |
| 364 | struct | MessageSchedule | data/schema coupling and match/serde break risk |
| 380 | impl | MessageSchedule | method behavior and trait semantics |
| 381 | fn | new | API and behavior coupling to callers/implementers |
| 404 | fn | should_fire | API and behavior coupling to callers/implementers |
| 417 | fn | build_message_matrix | API and behavior coupling to callers/implementers |
| 709 | struct | CanBus | data/schema coupling and match/serde break risk |
| 745 | impl | CanBus | method behavior and trait semantics |
| 746 | fn | new | API and behavior coupling to callers/implementers |
| 772 | fn | transmit | API and behavior coupling to callers/implementers |
| 777 | fn | transmit_tp | API and behavior coupling to callers/implementers |
| 793 | fn | tick | API and behavior coupling to callers/implementers |
| 920 | fn | log_error | API and behavior coupling to callers/implementers |
| 941 | fn | receive_for | API and behavior coupling to callers/implementers |
| 952 | struct | CanNetwork | data/schema coupling and match/serde break risk |
| 976 | struct | BusSnapshot | data/schema coupling and match/serde break risk |
| 987 | struct | CanNetworkSnapshot | data/schema coupling and match/serde break risk |
| 996 | impl | Default for CanNetwork | method behavior and trait semantics |
| 997 | fn | default | API and behavior coupling to callers/implementers |
| 1002 | impl | CanNetwork | method behavior and trait semantics |
| 1003 | fn | new | API and behavior coupling to callers/implementers |
| 1037 | fn | register | API and behavior coupling to callers/implementers |
| 1041 | fn | set_online | API and behavior coupling to callers/implementers |
| 1051 | fn | transmit | API and behavior coupling to callers/implementers |
| 1060 | fn | inject_error_once | API and behavior coupling to callers/implementers |
| 1073 | fn | inject_bus_off_once | API and behavior coupling to callers/implementers |
| 1077 | fn | clear_injections | API and behavior coupling to callers/implementers |
| 1105 | fn | tick | API and behavior coupling to callers/implementers |
| 1127 | fn | route_frames | API and behavior coupling to callers/implementers |
| 1147 | fn | bus_ref | API and behavior coupling to callers/implementers |
| 1158 | fn | bus_mut | API and behavior coupling to callers/implementers |
| 1170 | fn | receive_for_sa | API and behavior coupling to callers/implementers |
| 1180 | fn | all_errors | API and behavior coupling to callers/implementers |
| 1189 | fn | online_count | API and behavior coupling to callers/implementers |
| 1192 | fn | total_errors | API and behavior coupling to callers/implementers |
```

```
| 1199 | fn | per_bus_health_01 | API and behavior coupling to callers/implementers |
| 1212 | fn | network_health_score_01 | API and behavior coupling to callers/implementers |
| 1233 | fn | snapshot | API and behavior coupling to callers/implementers |
| 1264 | fn | export_snapshot_json | API and behavior coupling to callers/implementers |
| 1274 | fn | export_snapshot_csv | API and behavior coupling to callers/implementers |
| 1310 | module | tests | namespace and compile-time linkage |
| 1314 | fn | arbitration_orders_by_can_id | API and behavior coupling to callers/implementers |
| 1337 | fn | scheduler_jitter_stays_in_expected_band | API and behavior coupling to callers/implementers |
| 1363 | fn | routed_message_appears_next_tick_on_destination_bus | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! CAN Network — Multi-bus CAN simulation 1:1 with real heavy machinery. | comment/documentation; changing affec
| 2 | #![allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 4 | //! Real vehicles have multiple CAN buses with different speeds and purposes: | comment/documentation; changing a
| 5 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 6 | //!   BUS-1  HS-CAN Powertrain  500 kbps  ECM, TCM, ABS/ESP, VCM | comment/documentation; changing affects unders
| 7 | //!   BUS-2  HS-CAN Chassis     500 kbps  ABS, Steering, Suspension, Brakes | comment/documentation; changing affe
| 8 | //!   BUS-3  MS-CAN Body        250 kbps  BCM, ICM, PEPS, HVAC, Doors | comment/documentation; changing affects u
| 9 | //!   BUS-4  HS-CAN ISOBUS      250 kbps  HCM, Implement, TaskCtrl, VT | comment/documentation; changing affects
| 10 | //!   BUS-5  Diagnostic CAN     500 kbps  OBD-II port, UDS client connection | comment/documentation; changing a
| 11 | //!   BUS-6  LIN (master bus)    20 kbps  Simple sensors/actuators (seats, mirrors) | comment/documentation; char
| 12 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 13 | //! Each bus has independent: | comment/documentation; changing affects understanding and intent traceability. |
| 14 | //!   • Bit timing (prescaler, phase segments, SJW) | comment/documentation; changing affects understanding and
| 15 | //!   • Error counters (TEC/REC) per node | comment/documentation; changing affects understanding and intent tra
| 16 | //!   • Bus load calculation | comment/documentation; changing affects understanding and intent traceability. |
| 17 | //!   • Frame arbitration (priority-based) | comment/documentation; changing affects understanding and intent tra
| 18 | //!   • Error frame injection | comment/documentation; changing affects understanding and intent traceability. |
| 19 | //!   • CAN FD payload support (up to 64 bytes) | comment/documentation; changing affects understanding and inte
| 20 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 21 | //! J1939 Transport Protocol (TP) is implemented over BUS-1 and BUS-4: | comment/documentation; changing affects
| 22 | //!   BAM  — Broadcast Announce Message (multi-packet broadcast) | comment/documentation; changing affects unders
| 23 | //!   CMDT — Connection Mode Data Transfer (peer-to-peer with flow control) | comment/documentation; changing af
| 24 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 25 | //! The VCM (Vehicle Control Module) acts as gateway between buses, forwarding | comment/documentation; changing
| 26 | //! signals between powertrain, body, and ISOBUS domains. | comment/documentation; changing affects understanding
| 27 | (blank) | blank separator; no runtime behavior. |
| 28 | use crate::j1939::{addr, pgn, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 29 | use serde::Serialize; | import wiring; changing path/name can break compilation and module linkage. |
| 30 | use std::collections::{HashMap, VecDeque}; | import wiring; changing path/name can break compilation and module
| 31 | use std::fs; | import wiring; changing path/name can break compilation and module linkage. |
| 32 | use std::io::Write; | import wiring; changing path/name can break compilation and module linkage. |
| 33 | use std::path::Path; | import wiring; changing path/name can break compilation and module linkage. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 | // ■■ CAN Bus Speed ■■■
```

```
| 63 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 64 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 65 | (blank) | blank separator; no runtime behavior. |
| 66 | // ■■ Bus ID ■■■
```

```
| 139 |     pub kind: CanErrorKind, | support logic; impact depends on neighboring block and data flow. |
| 140 |     pub source_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 141 |     pub raw_id: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 142 |     pub description: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 143 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 144 | (blank) | blank separator; no runtime behavior. |
| 145 | // ■■ Registered node ■■■
```

```
| 215 | struct TpSession { | data layout contract; field changes can break callers/serde/tests. |
| 216 |     pgn: u32, | support logic; impact depends on neighboring block and data flow. |
| 217 |     sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 218 |     da: u8, // 0xFF = BAM broadcast | support logic; impact depends on neighboring block and data flow. |
| 219 |     data: Vec<u8>, | support logic; impact depends on neighboring block and data flow. |
| 220 |     total_bytes: u16, | support logic; impact depends on neighboring block and data flow. |
| 221 |     total_packets: u8, | support logic; impact depends on neighboring block and data flow. |
| 222 |     next_seq: u8, | support logic; impact depends on neighboring block and data flow. |
| 223 |     timer: f64, | support logic; impact depends on neighboring block and data flow. |
| 224 |     is_bam: bool, | support logic; impact depends on neighboring block and data flow. |
| 225 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 226 | (blank) | blank separator; no runtime behavior. |
| 227 | /// J1939 Transport Protocol engine (BAM + CMDT) | comment/documentation; changing affects understanding and int
| 228 | pub struct CanTransportProtocol { | data layout contract; field changes can break callers/serde/tests. |
| 229 |     /// Active send sessions: key = (sa, da, pgn) | comment/documentation; changing affects understanding and in
| 230 |     tx_sessions: Vec<TpSession>, | support logic; impact depends on neighboring block and data flow. |
| 231 |     /// Active receive sessions: key = (sa, da) | comment/documentation; changing affects understanding and inte
| 232 |     rx_sessions: Vec<TpSession>, | support logic; impact depends on neighboring block and data flow. |
| 233 |     /// Completed reassembled messages ready for delivery | comment/documentation; changing affects understandin
| 234 |     pub completed: Vec<(u32, u8, u8, Vec<u8>)>, // (pgn, sa, da, data) | support logic; impact depends on neighb
| 235 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 236 | (blank) | blank separator; no runtime behavior. |
| 237 | impl Default for CanTransportProtocol { | implementation binding; behavior semantics and trait conformance live
| 238 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 239 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 240 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 241 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 242 | (blank) | blank separator; no runtime behavior. |
| 243 | impl CanTransportProtocol { | implementation binding; behavior semantics and trait conformance live here. |
| 244 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 245 |         CanTransportProtocol { | support logic; impact depends on neighboring block and data flow. |
| 246 |             tx_sessions: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 247 |             rx_sessions: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 248 |             completed: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 249 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 250 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 251 | (blank) | blank separator; no runtime behavior. |
| 252 |     /// Queue a large J1939 message for BAM broadcast (> 8 bytes) | comment/documentation; changing affects unde
| 253 |     pub fn send_bam(&mut self, ts: f64, pgn: u32, sa: u8, data: Vec<u8>) -> Vec<J1939Frame> { | function signatu
| 254 |         let total = data.len() as u16; | support logic; impact depends on neighboring block and data flow. |
| 255 |         let packets = total.div_ceil(7) as u8; | support logic; impact depends on neighboring block and data flo
| 256 |         let mut frames = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 257 | (blank) | blank separator; no runtime behavior. |
| 258 |         // TP.CM_BAM — announce the broadcast | comment/documentation; changing affects understanding and intent
| 259 |         let mut bam = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 260 |         bam[0] = 0x20; // Control byte: BAM | support logic; impact depends on neighboring block and data flow.
| 261 |         bam[1] = (total & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 262 |         bam[2] = ((total >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow
| 263 |         bam[3] = packets; | support logic; impact depends on neighboring block and data flow. |
| 264 |         bam[4] = 0xFF; | support logic; impact depends on neighboring block and data flow. |
| 265 |         bam[5] = (pgn & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 266 |         bam[6] = ((pgn >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow.
| 267 |         bam[7] = ((pgn >> 16) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow
| 268 |         frames.push(J1939Frame::from_raw( | support logic; impact depends on neighboring block and data flow. |
| 269 |             ts, | support logic; impact depends on neighboring block and data flow. |
| 270 |             J1939Frame::build_id(7, pgn::TP_CM, sa, 0xFF), | support logic; impact depends on neighboring block
| 271 |             &bam, | support logic; impact depends on neighboring block and data flow. |
| 272 |         )); | support logic; impact depends on neighboring block and data flow. |
| 273 | (blank) | blank separator; no runtime behavior. |
| 274 |         // TP.DT — data transfer packets (send immediately for BAM) | comment/documentation; changing affects un
| 275 |         for seq in 0..packets { | iteration logic; may alter timing, throughput, and safety constraints. |
| 276 |             let start = seq as usize * 7; | support logic; impact depends on neighboring block and data flow. |
| 277 |             let end = (start + 7).min(data.len()); | support logic; impact depends on neighboring block and data
| 278 |             let mut dt = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 279 |             dt[0] = seq + 1; // sequence 1-based | support logic; impact depends on neighboring block and data
| 280 |             for (i, &b) in data[start..end].iter().enumerate() { | iteration logic; may alter timing, throughpu
| 281 |                 dt[i + 1] = b; | support logic; impact depends on neighboring block and data flow. |
| 282 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 283 |             frames.push(J1939Frame::from_raw( | support logic; impact depends on neighboring block and data flow
| 284 |                 ts + (seq as f64 * 0.010), | support logic; impact depends on neighboring block and data flow.
| 285 |                 J1939Frame::build_id(7, pgn::TP_DT, sa, 0xFF), | support logic; impact depends on neighboring b
| 286 |                 &dt, | support logic; impact depends on neighboring block and data flow. |
| 287 |             )); | support logic; impact depends on neighboring block and data flow. |
| 288 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 289 |         frames | support logic; impact depends on neighboring block and data flow. |
| 290 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 291 | (blank) | blank separator; no runtime behavior. |
| 292 |     /// Process received TP frame — reassemble multi-packet messages | comment/documentation; changing affects u
| 293 |     pub fn process_rx(&mut self, frame: &J1939Frame) -> bool { | function signature/body; changes affect API bel
| 294 |         match frame.pgn { | branch dispatcher; arm changes alter behavior class handling. |
| 295 |             p if p == pgn::TP_CM => { | support logic; impact depends on neighboring block and data flow. |
| 296 |                 let ctrl = frame.data[0]; | support logic; impact depends on neighboring block and data flow. |
| 297 |                 if ctrl == 0x20 { | runtime gate; condition changes can enable/disable critical paths. |
| 298 |                     // BAM announce | comment/documentation; changing affects understanding and intent traceabil
| 299 |                     let total = (frame.data[1] as u16) \| ((frame.data[2] as u16) << 8); | support logic; impact
| 300 |                     let packets = frame.data[3]; | support logic; impact depends on neighboring block and data f
| 301 |                     let pgn_rx = (frame.data[5] as u32) | support logic; impact depends on neighboring block and
| 302 |                         \| ((frame.data[6] as u32) << 8) | support logic; impact depends on neighboring block an
| 303 |                         \| ((frame.data[7] as u32) << 16); | support logic; impact depends on neighboring block
| 304 |                     self.rx_sessions.retain(\|s\| s.sa != frame.sa); | support logic; impact depends on neighbor
| 305 |                     self.rx_sessions.push(TpSession { | support logic; impact depends on neighboring block and d
| 306 |                         pgn: pgn_rx, | support logic; impact depends on neighboring block and data flow. |
| 307 |                         sa: frame.sa, | support logic; impact depends on neighboring block and data flow. |
| 308 |                         da: 0xFF, | support logic; impact depends on neighboring block and data flow. |
| 309 |                         data: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 310 |                         total_bytes: total, | support logic; impact depends on neighboring block and data flow.
| 311 |                         total_packets: packets, | support logic; impact depends on neighboring block and data fl
| 312 |                         next_seq: 1, | support logic; impact depends on neighboring block and data flow. |
| 313 |                         timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 314 |                         is_bam: true, | support logic; impact depends on neighboring block and data flow. |
| 315 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 316 |                     return true; | support logic; impact depends on neighboring block and data flow. |
| 317 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 318 |                 false | support logic; impact depends on neighboring block and data flow. |
| 319 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 320 |             p if p == pgn::TP_DT => { | support logic; impact depends on neighboring block and data flow. |
| 321 |                 let seq = frame.data[0]; | support logic; impact depends on neighboring block and data flow. |
| 322 |                 if let Some(sess) = self | runtime gate; condition changes can enable/disable critical paths. |
| 323 |                     .rx_sessions | support logic; impact depends on neighboring block and data flow. |
| 324 |                     .iter_mut() | support logic; impact depends on neighboring block and data flow. |
| 325 |                     .find(\|s\| s.sa == frame.sa && s.next_seq == seq) | support logic; impact depends on neighb
| 326 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 327 |                     let payload = &frame.data[1..]; | support logic; impact depends on neighboring block and dat
| 328 |                     let remaining = (sess.total_bytes as usize).saturating_sub(sess.data.len()); | support logic
| 329 |                     let to_copy = remaining.min(7); | support logic; impact depends on neighboring block and dat
| 330 |                     sess.data | support logic; impact depends on neighboring block and data flow. |
| 331 |                         .extend_from_slice(&payload[..to_copy.min(payload.len())]); | support logic; impact depe
| 332 |                     sess.next_seq += 1; | support logic; impact depends on neighboring block and data flow. |
| 333 |                     if sess.data.len() >= sess.total_bytes as usize { | runtime gate; condition changes can enal
| 334 |                         let done = sess.clone(); | support logic; impact depends on neighboring block and data f
| 335 |                         self.rx_sessions.retain(\|s\| s.sa != frame.sa); | support logic; impact depends on neig
| 336 |                         self.completed.push((done.pgn, done.sa, done.da, done.data)); | support logic; impact de
| 337 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 338 |                     return true; | support logic; impact depends on neighboring block and data flow. |
| 339 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 |                 false | support logic; impact depends on neighboring block and data flow. |
| 341 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 342 |             _ => false, | support logic; impact depends on neighboring block and data flow. |
| 343 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 344 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 345 | (blank) | blank separator; no runtime behavior. |
| 346 |     pub fn tick(&mut self, dt: f64) { | function signature/body; changes affect API behavior and control-flow.
| 347 |         for s in &mut self.tx_sessions { | iteration logic; may alter timing, throughput, and safety constraints
| 348 |             s.timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 349 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 350 |         for s in &mut self.rx_sessions { | iteration logic; may alter timing, throughput, and safety constraints
| 351 |             s.timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 352 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 353 |         // Timeout stale sessions after 750ms (J1939 specification) | comment/documentation; changing affects un
| 354 |         self.tx_sessions.retain(\|s\| s.timer < 0.750); | support logic; impact depends on neighboring block and
| 355 |         self.rx_sessions.retain(\|s\| s.timer < 0.750); | support logic; impact depends on neighboring block and
| 356 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 357 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 358 | (blank) | blank separator; no runtime behavior. |
| 359 | // ━━━
```

```
| 367 |     pub da: u8, // 0xFF = broadcast | support logic; impact depends on neighboring block and data flow. |
| 368 |     pub bus: BusId, | support logic; impact depends on neighboring block and data flow. |
| 369 |     pub period_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 370 |     pub priority: u8, | support logic; impact depends on neighboring block and data flow. |
| 371 |     /// Description for CAN database / DBC compatibility | comment/documentation; changing affects understanding
| 372 |     pub description: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 373 |     /// Current timer (fires when >= period_ms) | comment/documentation; changing affects understanding and inte
| 374 |     timer_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 375 |     /// Transmission jitter ±jitter_ms (realistic) | comment/documentation; changing affects understanding and i
| 376 |     pub jitter_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 377 |     jitter_offset: f64, | support logic; impact depends on neighboring block and data flow. |
| 378 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 379 | (blank) | blank separator; no runtime behavior. |
| 380 | impl MessageSchedule { | implementation binding; behavior semantics and trait conformance live here. |
| 381 |     pub fn new( | function signature/body; changes affect API behavior and control-flow. |
| 382 |         pgn: u32, | support logic; impact depends on neighboring block and data flow. |
| 383 |         sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 384 |         da: u8, | support logic; impact depends on neighboring block and data flow. |
| 385 |         bus: BusId, | support logic; impact depends on neighboring block and data flow. |
| 386 |         period_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 387 |         priority: u8, | support logic; impact depends on neighboring block and data flow. |
| 388 |         desc: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 389 |     ) -> Self { | support logic; impact depends on neighboring block and data flow. |
| 390 |         MessageSchedule { | support logic; impact depends on neighboring block and data flow. |
| 391 |             pgn, | support logic; impact depends on neighboring block and data flow. |
| 392 |             sa, | support logic; impact depends on neighboring block and data flow. |
| 393 |             da, | support logic; impact depends on neighboring block and data flow. |
| 394 |             bus, | support logic; impact depends on neighboring block and data flow. |
| 395 |             period_ms, | support logic; impact depends on neighboring block and data flow. |
| 396 |             priority, | support logic; impact depends on neighboring block and data flow. |
| 397 |             description: desc, | support logic; impact depends on neighboring block and data flow. |
| 398 |             timer_ms: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 399 |             jitter_ms: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 400 |             jitter_offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 401 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 402 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 403 |     /// Returns true if it's time to transmit, applying jitter | comment/documentation; changing affects understa
| 404 |     pub fn should_fire(&mut self, dt_ms: f64, noise: f64) -> bool { | function signature/body; changes affect AP
| 405 |         self.jitter_offset = noise * self.jitter_ms; | support logic; impact depends on neighboring block and da
| 406 |         self.timer_ms += dt_ms; | support logic; impact depends on neighboring block and data flow. |
| 407 |         if self.timer_ms >= self.period_ms + self.jitter_offset { | runtime gate; condition changes can enable/d
| 408 |             self.timer_ms = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 409 |             true | support logic; impact depends on neighboring block and data flow. |
| 410 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 411 |             false | support logic; impact depends on neighboring block and data flow. |
| 412 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 413 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 414 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 415 | (blank) | blank separator; no runtime behavior. |
| 416 | /// The complete J1939 message matrix for a heavy agricultural vehicle | comment/documentation; changing affects
| 417 | pub fn build_message_matrix() -> Vec<MessageSchedule> { | function signature/body; changes affect API behavior a
| 418 |     vec![ | support logic; impact depends on neighboring block and data flow. |
| 419 |         // ■ Powertrain bus (BUS-1 HS-CAN 500kbps) ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentation;
| 420 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 421 |             pgn::EEC1, | support logic; impact depends on neighboring block and data flow. |
| 422 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 423 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 424 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 425 |             10.0, | support logic; impact depends on neighboring block and data flow. |
| 426 |             3, | support logic; impact depends on neighboring block and data flow. |
| 427 |             "EEC1: Engine Speed/Torque", | support logic; impact depends on neighboring block and data flow. |
| 428 |         ), | support logic; impact depends on neighboring block and data flow. |
| 429 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 430 |             pgn::EEC2, | support logic; impact depends on neighboring block and data flow. |
| 431 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 432 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 433 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 434 |             50.0, | support logic; impact depends on neighboring block and data flow. |
| 435 |             3, | support logic; impact depends on neighboring block and data flow. |
| 436 |             "EEC2: Throttle/Load", | support logic; impact depends on neighboring block and data flow. |
| 437 |         ), | support logic; impact depends on neighboring block and data flow. |
| 438 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 439 |             pgn::IC1, | support logic; impact depends on neighboring block and data flow. |
| 440 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 441 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 442 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
```

```
| 443 |             500.0, | support logic; impact depends on neighboring block and data flow. |
| 444 |             6, | support logic; impact depends on neighboring block and data flow. |
| 445 |             "IC1: Boost/Intake Conditions", | support logic; impact depends on neighboring block and data flow.
| 446 |         ), | support logic; impact depends on neighboring block and data flow. |
| 447 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 448 |             pgn::ET1, | support logic; impact depends on neighboring block and data flow. |
| 449 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 450 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 451 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 452 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 453 |             6, | support logic; impact depends on neighboring block and data flow. |
| 454 |             "ET1: Engine Temperatures", | support logic; impact depends on neighboring block and data flow. |
| 455 |         ), | support logic; impact depends on neighboring block and data flow. |
| 456 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 457 |             pgn::EFL_P1, | support logic; impact depends on neighboring block and data flow. |
| 458 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 459 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 460 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 461 |             500.0, | support logic; impact depends on neighboring block and data flow. |
| 462 |             6, | support logic; impact depends on neighboring block and data flow. |
| 463 |             "EFL/P1: Oil/Fuel Pressures", | support logic; impact depends on neighboring block and data flow. |
| 464 |         ), | support logic; impact depends on neighboring block and data flow. |
| 465 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 466 |             pgn::LFE, | support logic; impact depends on neighboring block and data flow. |
| 467 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 468 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 469 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 470 |             100.0, | support logic; impact depends on neighboring block and data flow. |
| 471 |             6, | support logic; impact depends on neighboring block and data flow. |
| 472 |             "LFE: Fuel Economy", | support logic; impact depends on neighboring block and data flow. |
| 473 |         ), | support logic; impact depends on neighboring block and data flow. |
| 474 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 475 |             pgn::HOURS, | support logic; impact depends on neighboring block and data flow. |
| 476 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 477 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 478 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 479 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 480 |             6, | support logic; impact depends on neighboring block and data flow. |
| 481 |             "HOURS: Engine Hours", | support logic; impact depends on neighboring block and data flow. |
| 482 |         ), | support logic; impact depends on neighboring block and data flow. |
| 483 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 484 |             pgn::DM1, | support logic; impact depends on neighboring block and data flow. |
| 485 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 486 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 487 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 488 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 489 |             6, | support logic; impact depends on neighboring block and data flow. |
| 490 |             "DM1: Active DTCs (ECM)", | support logic; impact depends on neighboring block and data flow. |
| 491 |         ), | support logic; impact depends on neighboring block and data flow. |
| 492 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 493 |             pgn::EEC3, | support logic; impact depends on neighboring block and data flow. |
| 494 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 495 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 496 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 497 |             250.0, | support logic; impact depends on neighboring block and data flow. |
| 498 |             6, | support logic; impact depends on neighboring block and data flow. |
| 499 |             "EEC3: Engine Demand/Nominal", | support logic; impact depends on neighboring block and data flow.
| 500 |         ), | support logic; impact depends on neighboring block and data flow. |
| 501 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 502 |             pgn::FUEL1, | support logic; impact depends on neighboring block and data flow. |
| 503 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 504 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 505 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 506 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 507 |             6, | support logic; impact depends on neighboring block and data flow. |
| 508 |             "FUEL1: Cumulative Fuel", | support logic; impact depends on neighboring block and data flow. |
| 509 |         ), | support logic; impact depends on neighboring block and data flow. |
| 510 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 511 |             0, | support logic; impact depends on neighboring block and data flow. |
| 512 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 513 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 514 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 515 |             20.0, | support logic; impact depends on neighboring block and data flow. |
| 516 |             3, | support logic; impact depends on neighboring block and data flow. |
| 517 |             "TSC1: Torque/Speed Ctrl → TCM", | support logic; impact depends on neighboring block and data flow
| 518 |         ), | support logic; impact depends on neighboring block and data flow. |
```

```
| 519 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 520 |             pgn::ETC1, | support logic; impact depends on neighboring block and data flow. |
| 521 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 522 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 523 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 524 |             20.0, | support logic; impact depends on neighboring block and data flow. |
| 525 |             3, | support logic; impact depends on neighboring block and data flow. |
| 526 |             "ETC1: Trans Output Shaft Speed", | support logic; impact depends on neighboring block and data flow
| 527 |         ), | support logic; impact depends on neighboring block and data flow. |
| 528 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 529 |             pgn::ETC2, | support logic; impact depends on neighboring block and data flow. |
| 530 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 531 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 532 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 533 |             20.0, | support logic; impact depends on neighboring block and data flow. |
| 534 |             3, | support logic; impact depends on neighboring block and data flow. |
| 535 |             "ETC2: Gear/Range Position", | support logic; impact depends on neighboring block and data flow. |
| 536 |         ), | support logic; impact depends on neighboring block and data flow. |
| 537 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 538 |             pgn::DM1, | support logic; impact depends on neighboring block and data flow. |
| 539 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 540 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 541 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 542 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 543 |             6, | support logic; impact depends on neighboring block and data flow. |
| 544 |             "DM1: Active DTCs (TCM)", | support logic; impact depends on neighboring block and data flow. |
| 545 |         ), | support logic; impact depends on neighboring block and data flow. |
| 546 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 547 |             pgn::TCFG, | support logic; impact depends on neighboring block and data flow. |
| 548 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 549 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 550 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 551 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 552 |             7, | support logic; impact depends on neighboring block and data flow. |
| 553 |             "TCFG: Transmission Config", | support logic; impact depends on neighboring block and data flow. |
| 554 |         ), | support logic; impact depends on neighboring block and data flow. |
| 555 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 556 |             pgn::EBC1, | support logic; impact depends on neighboring block and data flow. |
| 557 |             addr::BRAKES, | support logic; impact depends on neighboring block and data flow. |
| 558 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 559 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 560 |             20.0, | support logic; impact depends on neighboring block and data flow. |
| 561 |             2, | support logic; impact depends on neighboring block and data flow. |
| 562 |             "EBC1: Brake Status (ABS/ESP)", | support logic; impact depends on neighboring block and data flow.
| 563 |         ), | support logic; impact depends on neighboring block and data flow. |
| 564 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 565 |             pgn::EBC2, | support logic; impact depends on neighboring block and data flow. |
| 566 |             addr::BRAKES, | support logic; impact depends on neighboring block and data flow. |
| 567 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 568 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 569 |             100.0, | support logic; impact depends on neighboring block and data flow. |
| 570 |             6, | support logic; impact depends on neighboring block and data flow. |
| 571 |             "EBC2: Wheel Speeds", | support logic; impact depends on neighboring block and data flow. |
| 572 |         ), | support logic; impact depends on neighboring block and data flow. |
| 573 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 574 |             pgn::DM1, | support logic; impact depends on neighboring block and data flow. |
| 575 |             addr::BRAKES, | support logic; impact depends on neighboring block and data flow. |
| 576 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 577 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 578 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 579 |             6, | support logic; impact depends on neighboring block and data flow. |
| 580 |             "DM1: Active DTCs (ABS)", | support logic; impact depends on neighboring block and data flow. |
| 581 |         ), | support logic; impact depends on neighboring block and data flow. |
| 582 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 583 |             pgn::CCVS, | support logic; impact depends on neighboring block and data flow. |
| 584 |             addr::INSTRUMENT, | support logic; impact depends on neighboring block and data flow. |
| 585 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 586 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 587 |             100.0, | support logic; impact depends on neighboring block and data flow. |
| 588 |             6, | support logic; impact depends on neighboring block and data flow. |
| 589 |             "CCVS: Vehicle Speed/CC", | support logic; impact depends on neighboring block and data flow. |
| 590 |         ), | support logic; impact depends on neighboring block and data flow. |
| 591 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 592 |             pgn::VD, | support logic; impact depends on neighboring block and data flow. |
| 593 |             addr::INSTRUMENT, | support logic; impact depends on neighboring block and data flow. |
| 594 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
```

```
| 595 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 596 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 597 |             6, | support logic; impact depends on neighboring block and data flow. |
| 598 |             "VD: Vehicle Distance", | support logic; impact depends on neighboring block and data flow. |
| 599 |         ), | support logic; impact depends on neighboring block and data flow. |
| 600 |         // ■ ISOBUS (BUS-4, 250kbps) ■■■
```

```
| 671 |             6, | support logic; impact depends on neighboring block and data flow. |
| 672 |             "VEP1: Vehicle Electrical Power", | support logic; impact depends on neighboring block and data flow
| 673 |         ), | support logic; impact depends on neighboring block and data flow. |
| 674 |         // ■ Network Management (all buses) ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentati
| 675 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 676 |             pgn::ECAN, | support logic; impact depends on neighboring block and data flow. |
| 677 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 678 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 679 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 680 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 681 |             6, | support logic; impact depends on neighboring block and data flow. |
| 682 |             "AC: Address Claim (ECM)", | support logic; impact depends on neighboring block and data flow. |
| 683 |         ), | support logic; impact depends on neighboring block and data flow. |
| 684 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 685 |             pgn::ECAN, | support logic; impact depends on neighboring block and data flow. |
| 686 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 687 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 688 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 689 |             1000.0, | support logic; impact depends on neighboring block and data flow. |
| 690 |             6, | support logic; impact depends on neighboring block and data flow. |
| 691 |             "AC: Address Claim (TCM)", | support logic; impact depends on neighboring block and data flow. |
| 692 |         ), | support logic; impact depends on neighboring block and data flow. |
| 693 |         MessageSchedule::new( | support logic; impact depends on neighboring block and data flow. |
| 694 |             pgn::DM13, | support logic; impact depends on neighboring block and data flow. |
| 695 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 696 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 697 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 698 |             5000.0, | support logic; impact depends on neighboring block and data flow. |
| 699 |             7, | support logic; impact depends on neighboring block and data flow. |
| 700 |             "DM13: Stop/Start Broadcast", | support logic; impact depends on neighboring block and data flow. |
| 701 |         ), | support logic; impact depends on neighboring block and data flow. |
| 702 |     ] | support logic; impact depends on neighboring block and data flow. |
| 703 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 704 | (blank) | blank separator; no runtime behavior. |
| 705 | // ■■■■■■■■■■■■■■■■■■■
```

```
| 747 |         CanBus { | support logic; impact depends on neighboring block and data flow. |
| 748 |             id, | support logic; impact depends on neighboring block and data flow. |
| 749 |             speed, | support logic; impact depends on neighboring block and data flow. |
| 750 |             state: BusState::ErrorActive, | support logic; impact depends on neighboring block and data flow. |
| 751 |             can_fd: false, | support logic; impact depends on neighboring block and data flow. |
| 752 |             tx_queue: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 753 |             dispatched: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 754 |             frame_log: VecDeque::new(), | support logic; impact depends on neighboring block and data flow. |
| 755 |             total_tx: 0, | support logic; impact depends on neighboring block and data flow. |
| 756 |             total_errors: 0, | support logic; impact depends on neighboring block and data flow. |
| 757 |             bus_load_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 758 |             frame_counter: 0, | support logic; impact depends on neighboring block and data flow. |
| 759 |             load_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 760 |             inject_bit_error: false, | support logic; impact depends on neighboring block and data flow. |
| 761 |             inject_bus_off: false, | support logic; impact depends on neighboring block and data flow. |
| 762 |             inject_missing_ack: false, | support logic; impact depends on neighboring block and data flow. |
| 763 |             inject_babbling_sa: None, | support logic; impact depends on neighboring block and data flow. |
| 764 |             busoff_recovery_counter: 0, | support logic; impact depends on neighboring block and data flow. |
| 765 |             busoff_recovery_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 766 |             error_log: VecDeque::new(), | support logic; impact depends on neighboring block and data flow. |
| 767 |             tp: CanTransportProtocol::new(), | support logic; impact depends on neighboring block and data flow
| 768 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 769 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 770 | (blank) | blank separator; no runtime behavior. |
| 771 |     /// Enqueue a frame for transmission | comment/documentation; changing affects understanding and intent trac
| 772 |     pub fn transmit(&mut self, frame: J1939Frame) { | function signature/body; changes affect API behavior and c
| 773 |         self.tx_queue.push(frame); | support logic; impact depends on neighboring block and data flow. |
| 774 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 775 | (blank) | blank separator; no runtime behavior. |
| 776 |     /// Transmit a large message via TP.BAM | comment/documentation; changing affects understanding and intent t
| 777 |     pub fn transmit_tp(&mut self, ts: f64, pgn: u32, sa: u8, data: Vec<u8>) { | function signature/body; changes
| 778 |         if data.len() <= 8 { | runtime gate; condition changes can enable/disable critical paths. |
| 779 |             // Fits in one frame | comment/documentation; changing affects understanding and intent traceability
| 780 |             let id = J1939Frame::build_id(6, pgn, sa, 0xFF); | support logic; impact depends on neighboring bloc
| 781 |             let mut arr = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 782 |             arr[..data.len()].copy_from_slice(&data); | support logic; impact depends on neighboring block and c
| 783 |             self.transmit(J1939Frame::from_raw(ts, id, &arr)); | support logic; impact depends on neighboring bl
| 784 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 785 |             let frames = self.tp.send_bam(ts, pgn, sa, data); | protocol or I/O critical; changes may impact ext
| 786 |             for f in frames { | iteration logic; may alter timing, throughput, and safety constraints. |
| 787 |                 self.transmit(f); | support logic; impact depends on neighboring block and data flow. |
| 788 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 789 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 790 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 791 | (blank) | blank separator; no runtime behavior. |
| 792 |     /// Arbitrate and dispatch all queued frames for this cycle. | comment/documentation; changing affects under
| 793 |     pub fn tick(&mut self, nodes: &mut HashMap<u8, CanNode>, dt: f64) { | function signature/body; changes affec
| 794 |         self.dispatched.clear(); | support logic; impact depends on neighboring block and data flow. |
| 795 |         self.tp.tick(dt); | support logic; impact depends on neighboring block and data flow. |
| 796 | (blank) | blank separator; no runtime behavior. |
| 797 |         // ■ Bus-off recovery timer ■■■
```

```
| 823 |             self.log_error( | support logic; impact depends on neighboring block and data flow. |
| 824 |                 0.0, | support logic; impact depends on neighboring block and data flow. |
| 825 |                 CanErrorKind::Bit, | support logic; impact depends on neighboring block and data flow. |
| 826 |                 None, | support logic; impact depends on neighboring block and data flow. |
| 827 |                 None, | support logic; impact depends on neighboring block and data flow. |
| 828 |                 "INJECTED: Bus-Off condition", | support logic; impact depends on neighboring block and data flo
| 829 |             ); | support logic; impact depends on neighboring block and data flow. |
| 830 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 831 | (blank) | blank separator; no runtime behavior. |
| 832 |         // ■ Babbling-idiot detection ■■■
```

```
| 899 |             self.total_tx += 1; | support logic; impact depends on neighboring block and data flow. |
| 900 |             self.frame_counter += 1; | support logic; impact depends on neighboring block and data flow. |
| 901 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 902 | (blank) | blank separator; no runtime behavior. |
| 903 |         // ■ Bus load (updated each second) ■■■
```

```
| 975 | #[derive(Debug, Clone, Serialize)] | comment/documentation; changing affects understanding and intent traceabili
| 976 | pub struct BusSnapshot { | data layout contract; field changes can break callers/serde/tests. |
| 977 |     pub bus: String, | support logic; impact depends on neighboring block and data flow. |
| 978 |     pub speed_kbps: u32, | support logic; impact depends on neighboring block and data flow. |
| 979 |     pub state: String, | support logic; impact depends on neighboring block and data flow. |
| 980 |     pub load_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 981 |     pub total_tx: u64, | support logic; impact depends on neighboring block and data flow. |
| 982 |     pub total_errors: u64, | support logic; impact depends on neighboring block and data flow. |
| 983 |     pub log_depth: usize, | support logic; impact depends on neighboring block and data flow. |
| 984 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 985 | (blank) | blank separator; no runtime behavior. |
| 986 | #[derive(Debug, Clone, Serialize)] | comment/documentation; changing affects understanding and intent traceabili
| 987 | pub struct CanNetworkSnapshot { | data layout contract; field changes can break callers/serde/tests. |
| 988 |     pub elapsed_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 989 |     pub health_score_01: f64, | support logic; impact depends on neighboring block and data flow. |
| 990 |     pub total_frames_all_buses: u64, | support logic; impact depends on neighboring block and data flow. |
| 991 |     pub total_errors_all_buses: u64, | support logic; impact depends on neighboring block and data flow. |
| 992 |     pub online_nodes: usize, | support logic; impact depends on neighboring block and data flow. |
| 993 |     pub buses: Vec<BusSnapshot>, | support logic; impact depends on neighboring block and data flow. |
| 994 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 995 | (blank) | blank separator; no runtime behavior. |
| 996 | impl Default for CanNetwork { | implementation binding; behavior semantics and trait conformance live here. |
| 997 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 998 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 999 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1000 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1001 | (blank) | blank separator; no runtime behavior. |
| 1002 | impl CanNetwork { | implementation binding; behavior semantics and trait conformance live here. |
| 1003 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 1004 |         let mut net = CanNetwork { | support logic; impact depends on neighboring block and data flow. |
| 1005 |             powertrain: CanBus::new(BusId::PowertrainHs, CanSpeed::Kbps500), | support logic; impact depends o
| 1006 |             chassis: CanBus::new(BusId::ChassisHs, CanSpeed::Kbps500), | support logic; impact depends on neigh
| 1007 |             body: CanBus::new(BusId::BodyMs, CanSpeed::Kbps250), | support logic; impact depends on neighboring
| 1008 |             isobus: CanBus::new(BusId::IsoBus, CanSpeed::Kbps250), | support logic; impact depends on neighbori
| 1009 |             diagnostic: CanBus::new(BusId::Diagnostic, CanSpeed::Kbps500), | support logic; impact depends on n
| 1010 |             nodes: HashMap::new(), | support logic; impact depends on neighboring block and data flow. |
| 1011 |             schedule: build_message_matrix(), | support logic; impact depends on neighboring block and data flo
| 1012 |             routing: vec![ | support logic; impact depends on neighboring block and data flow. |
| 1013 |                 // VCM forwards engine speed from powertrain to body bus (for speedometer over MS-CAN) | comme
| 1014 |                 (pgn::EEC1, BusId::PowertrainHs, BusId::BodyMs), | support logic; impact depends on neighboring
| 1015 |                 (pgn::CCVS, BusId::PowertrainHs, BusId::BodyMs), | support logic; impact depends on neighboring
| 1016 |                 (pgn::DM1, BusId::PowertrainHs, BusId::Diagnostic), | support logic; impact depends on neighbo
| 1017 |                 (pgn::ETC1, BusId::PowertrainHs, BusId::BodyMs), | support logic; impact depends on neighboring
| 1018 |             ], | support logic; impact depends on neighboring block and data flow. |
| 1019 |             total_frames_all_buses: 0, | support logic; impact depends on neighboring block and data flow. |
| 1020 |             elapsed: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1021 |             noise_t: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1022 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1023 |         // Register all known nodes on their respective buses | comment/documentation; changing affects unders
| 1024 |         net.register(addr::ECM_1, "ECM #1", BusId::PowertrainHs); | support logic; impact depends on neighborin
| 1025 |         net.register(addr::TRANSMISSION, "TCM", BusId::PowertrainHs); | support logic; impact depends on neigh
| 1026 |         net.register(addr::BRAKES, "ABS/ESP", BusId::PowertrainHs); | support logic; impact depends on neighbo
| 1027 |         net.register(addr::CAB, "BCM/CAB", BusId::BodyMs); | support logic; impact depends on neighboring bloc
| 1028 |         net.register(addr::INSTRUMENT, "ICM/DASH", BusId::BodyMs); | support logic; impact depends on neighbori
| 1029 |         net.register(addr::HITCH, "HCM/HITCH", BusId::IsoBus); | support logic; impact depends on neighboring b
| 1030 |         net.register(addr::ISOBUS_VT, "ISOBUS-VT", BusId::IsoBus); | support logic; impact depends on neighbori
| 1031 |         net.register(addr::TASK_CTRL, "ISOBUS-TC", BusId::IsoBus); | support logic; impact depends on neighbori
| 1032 |         net.register(addr::IMPLEMENT, "IMPLEMENT", BusId::IsoBus); | support logic; impact depends on neighbori
| 1033 |         net.register(addr::HEADWAY, "VCM/GW", BusId::PowertrainHs); | support logic; impact depends on neighbo
| 1034 |         net | support logic; impact depends on neighboring block and data flow. |
| 1035 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1036 | (blank) | blank separator; no runtime behavior. |
| 1037 |     pub fn register(&mut self, sa: u8, name: &'static str, bus: BusId) { | function signature/body; changes af
| 1038 |         self.nodes.insert(sa, CanNode::new(sa, name, bus)); | support logic; impact depends on neighboring bloc
| 1039 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1040 | (blank) | blank separator; no runtime behavior. |
| 1041 |     pub fn set_online(&mut self, sa: u8) { | function signature/body; changes affect API behavior and control-
| 1042 |         if let Some(n) = self.nodes.get_mut(&sa) { | runtime gate; condition changes can enable/disable critica
| 1043 |             n.online = true; | support logic; impact depends on neighboring block and data flow. |
| 1044 |             n.tec = 0; | support logic; impact depends on neighboring block and data flow. |
| 1045 |             n.rec = 0; | support logic; impact depends on neighboring block and data flow. |
| 1046 |             n.state = BusState::ErrorActive; | support logic; impact depends on neighboring block and data flow
| 1047 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1048 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1049 | (blank) | blank separator; no runtime behavior. |
| 1050 |     /// Transmit a frame on the appropriate bus for the SA | comment/documentation; changing affects understan
```

```
| 1051 |     pub fn transmit(&mut self, frame: J1939Frame) { | function signature/body; changes affect API behavior and
| 1052 |         let bus_id = self | support logic; impact depends on neighboring block and data flow. |
| 1053 |             .nodes | support logic; impact depends on neighboring block and data flow. |
| 1054 |             .get(&frame.sa) | support logic; impact depends on neighboring block and data flow. |
| 1055 |             .map(\|n\| n.bus) | support logic; impact depends on neighboring block and data flow. |
| 1056 |             .unwrap_or(BusId::PowertrainHs); | support logic; impact depends on neighboring block and data flow
| 1057 |         self.bus_mut(bus_id).transmit(frame); | support logic; impact depends on neighboring block and data flo
| 1058 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1059 | (blank) | blank separator; no runtime behavior. |
| 1060 |     pub fn inject_error_once(&mut self, bus: BusId, kind: CanErrorKind) { | function signature/body; changes af
| 1061 |         let babbling_sa = self.nodes.values().find(\|n\| n.bus == bus).map(\|n\| n.sa); | support logic; impact
| 1062 |         let b = self.bus_mut(bus); | support logic; impact depends on neighboring block and data flow. |
| 1063 |         match kind { | branch dispatcher; arm changes alter behavior class handling. |
| 1064 |             CanErrorKind::Bit => b.inject_bit_error = true, | support logic; impact depends on neighboring bloc
| 1065 |             CanErrorKind::Ack => b.inject_missing_ack = true, | support logic; impact depends on neighboring bl
| 1066 |             CanErrorKind::BablingIdiot => { | support logic; impact depends on neighboring block and data flow
| 1067 |                 b.inject_babbling_sa = babbling_sa; | support logic; impact depends on neighboring block and da
| 1068 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1069 |             _ => b.inject_bit_error = true, | support logic; impact depends on neighboring block and data flow
| 1070 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1071 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1072 | (blank) | blank separator; no runtime behavior. |
| 1073 |     pub fn inject_bus_off_once(&mut self, bus: BusId) { | function signature/body; changes affect API behavior
| 1074 |         self.bus_mut(bus).inject_bus_off = true; | support logic; impact depends on neighboring block and data
| 1075 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1076 | (blank) | blank separator; no runtime behavior. |
| 1077 |     pub fn clear_injections(&mut self, bus: Option<BusId>) { | function signature/body; changes affect API beha
| 1078 |         match bus { | branch dispatcher; arm changes alter behavior class handling. |
| 1079 |             Some(id) => { | support logic; impact depends on neighboring block and data flow. |
| 1080 |                 let b = self.bus_mut(id); | support logic; impact depends on neighboring block and data flow.
| 1081 |                 b.inject_bit_error = false; | support logic; impact depends on neighboring block and data flow
| 1082 |                 b.inject_missing_ack = false; | support logic; impact depends on neighboring block and data flo
| 1083 |                 b.inject_bus_off = false; | support logic; impact depends on neighboring block and data flow.
| 1084 |                 b.inject_babbling_sa = None; | support logic; impact depends on neighboring block and data flow
| 1085 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1086 |             None => { | support logic; impact depends on neighboring block and data flow. |
| 1087 |                 for id in [ | iteration logic; may alter timing, throughput, and safety constraints. |
| 1088 |                     BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 1089 |                     BusId::ChassisHs, | support logic; impact depends on neighboring block and data flow. |
| 1090 |                     BusId::BodyMs, | support logic; impact depends on neighboring block and data flow. |
| 1091 |                     BusId::IsoBus, | support logic; impact depends on neighboring block and data flow. |
| 1092 |                     BusId::Diagnostic, | support logic; impact depends on neighboring block and data flow. |
| 1093 |                 ] { | support logic; impact depends on neighboring block and data flow. |
| 1094 |                     let b = self.bus_mut(id); | support logic; impact depends on neighboring block and data flo
| 1095 |                     b.inject_bit_error = false; | support logic; impact depends on neighboring block and data
| 1096 |                     b.inject_missing_ack = false; | support logic; impact depends on neighboring block and data
| 1097 |                     b.inject_bus_off = false; | support logic; impact depends on neighboring block and data flo
| 1098 |                     b.inject_babbling_sa = None; | support logic; impact depends on neighboring block and data
| 1099 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1100 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1101 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1102 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1103 | (blank) | blank separator; no runtime behavior. |
| 1104 |     /// Tick all buses and collect stats | comment/documentation; changing affects understanding and intent tra
| 1105 |     pub fn tick(&mut self, dt: f64) { | function signature/body; changes affect API behavior and control-flow.
| 1106 |         self.elapsed += dt; | support logic; impact depends on neighboring block and data flow. |
| 1107 |         self.noise_t += dt; | support logic; impact depends on neighboring block and data flow. |
| 1108 |         let _noise = ((self.noise_t * 127.1 + 311.7).sin() * 43758.5).fract() - 0.5; | support logic; impact de
| 1109 | (blank) | blank separator; no runtime behavior. |
| 1110 |         // Tick each bus | comment/documentation; changing affects understanding and intent traceability. |
| 1111 |         self.powertrain.tick(&mut self.nodes, dt); | support logic; impact depends on neighboring block and dat
| 1112 |         self.chassis.tick(&mut self.nodes, dt); | support logic; impact depends on neighboring block and data
| 1113 |         self.body.tick(&mut self.nodes, dt); | support logic; impact depends on neighboring block and data flow
| 1114 |         self.isobus.tick(&mut self.nodes, dt); | support logic; impact depends on neighboring block and data f
| 1115 |         self.diagnostic.tick(&mut self.nodes, dt); | support logic; impact depends on neighboring block and dat
| 1116 | (blank) | blank separator; no runtime behavior. |
| 1117 |         // VCM routing: forward frames between buses | comment/documentation; changing affects understanding an
| 1118 |         self.route_frames(); | support logic; impact depends on neighboring block and data flow. |
| 1119 | (blank) | blank separator; no runtime behavior. |
| 1120 |         self.total_frames_all_buses = self.powertrain.total_tx | support logic; impact depends on neighboring b
| 1121 |             + self.chassis.total_tx | support logic; impact depends on neighboring block and data flow. |
| 1122 |             + self.body.total_tx | support logic; impact depends on neighboring block and data flow. |
| 1123 |             + self.isobus.total_tx | support logic; impact depends on neighboring block and data flow. |
| 1124 |             + self.diagnostic.total_tx; | support logic; impact depends on neighboring block and data flow. |
| 1125 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1126 | (blank) | blank separator; no runtime behavior. |
```

```
| 1127 |     fn route_frames(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1128 |         // Collect frames to route (can't borrow self.powertrain and self.body mutably simultaneously) | commen
| 1129 |         let mut to_route: Vec<(J1939Frame, BusId)> = Vec::new(); | support logic; impact depends on neighboring
| 1130 |         for (pgn_filter, src_bus, dst_bus) in &self.routing { | iteration logic; may alter timing, throughput,
| 1131 |             let src_frames: Vec<J1939Frame> = self | support logic; impact depends on neighboring block and da
| 1132 |                 .bus_ref(*src_bus) | support logic; impact depends on neighboring block and data flow. |
| 1133 |                 .dispatched | support logic; impact depends on neighboring block and data flow. |
| 1134 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 1135 |                 .filter(\|f\| f.pgn == *pgn_filter) | support logic; impact depends on neighboring block and da
| 1136 |                 .cloned() | support logic; impact depends on neighboring block and data flow. |
| 1137 |                 .collect(); | support logic; impact depends on neighboring block and data flow. |
| 1138 |             for f in src_frames { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1139 |                 to_route.push((f, *dst_bus)); | support logic; impact depends on neighboring block and data flo
| 1140 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1141 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1142 |         for (f, dst) in to_route { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1143 |             self.bus_mut(dst).transmit(f); | support logic; impact depends on neighboring block and data flow.
| 1144 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1145 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1146 | (blank) | blank separator; no runtime behavior. |
| 1147 |     fn bus_ref(&self, id: BusId) -> &CanBus { | function signature/body; changes affect API behavior and contro
| 1148 |         match id { | branch dispatcher; arm changes alter behavior class handling. |
| 1149 |             BusId::PowertrainHs => &self.powertrain, | support logic; impact depends on neighboring block and
| 1150 |             BusId::ChassisHs => &self.chassis, | support logic; impact depends on neighboring block and data f
| 1151 |             BusId::BodyMs => &self.body, | support logic; impact depends on neighboring block and data flow. |
| 1152 |             BusId::IsoBus => &self.isobus, | support logic; impact depends on neighboring block and data flow.
| 1153 |             BusId::Diagnostic => &self.diagnostic, | support logic; impact depends on neighboring block and da
| 1154 |             BusId::Lin => &self.body, // LIN master reuses body bus object | support logic; impact depends on
| 1155 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1156 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1157 | (blank) | blank separator; no runtime behavior. |
| 1158 |     fn bus_mut(&mut self, id: BusId) -> &mut CanBus { | function signature/body; changes affect API behavior a
| 1159 |         match id { | branch dispatcher; arm changes alter behavior class handling. |
| 1160 |             BusId::PowertrainHs => &mut self.powertrain, | support logic; impact depends on neighboring block
| 1161 |             BusId::ChassisHs => &mut self.chassis, | support logic; impact depends on neighboring block and da
| 1162 |             BusId::BodyMs => &mut self.body, | support logic; impact depends on neighboring block and data flo
| 1163 |             BusId::IsoBus => &mut self.isobus, | support logic; impact depends on neighboring block and data f
| 1164 |             BusId::Diagnostic => &mut self.diagnostic, | support logic; impact depends on neighboring block an
| 1165 |             BusId::Lin => &mut self.body, | support logic; impact depends on neighboring block and data flow.
| 1166 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1167 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1168 | (blank) | blank separator; no runtime behavior. |
| 1169 |     /// All frames from all buses received this cycle for a given SA | comment/documentation; changing affects
| 1170 |     pub fn receive_for_sa(&self, sa: u8) -> Vec<&J1939Frame> { | function signature/body; changes affect API b
| 1171 |         let bus_id = self | support logic; impact depends on neighboring block and data flow. |
| 1172 |             .nodes | support logic; impact depends on neighboring block and data flow. |
| 1173 |             .get(&sa) | support logic; impact depends on neighboring block and data flow. |
| 1174 |             .map(\|n\| n.bus) | support logic; impact depends on neighboring block and data flow. |
| 1175 |             .unwrap_or(BusId::PowertrainHs); | support logic; impact depends on neighboring block and data flo
| 1176 |         self.bus_ref(bus_id).receive_for(sa).collect() | support logic; impact depends on neighboring block an
| 1177 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1178 | (blank) | blank separator; no runtime behavior. |
| 1179 |     /// Combined error log from all buses | comment/documentation; changing affects understanding and intent t
| 1180 |     pub fn all_errors(&self) -> impl Iterator<Item = &BusError> { | function signature/body; changes affect AP
| 1181 |         self.powertrain | support logic; impact depends on neighboring block and data flow. |
| 1182 |             .error_log | support logic; impact depends on neighboring block and data flow. |
| 1183 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1184 |             .chain(self.chassis.error_log.iter()) | support logic; impact depends on neighboring block and data
| 1185 |             .chain(self.body.error_log.iter()) | support logic; impact depends on neighboring block and data f
| 1186 |             .chain(self.isobus.error_log.iter()) | support logic; impact depends on neighboring block and data
| 1187 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1188 | (blank) | blank separator; no runtime behavior. |
| 1189 |     pub fn online_count(&self) -> usize { | function signature/body; changes affect API behavior and control-f
| 1190 |         self.nodes.values().filter(\|n\| n.online).count() | support logic; impact depends on neighboring bloc
| 1191 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1192 |     pub fn total_errors(&self) -> u64 { | function signature/body; changes affect API behavior and control-flo
| 1193 |         self.powertrain.total_errors | support logic; impact depends on neighboring block and data flow. |
| 1194 |             + self.chassis.total_errors | support logic; impact depends on neighboring block and data flow. |
| 1195 |             + self.body.total_errors | support logic; impact depends on neighboring block and data flow. |
| 1196 |             + self.isobus.total_errors | support logic; impact depends on neighboring block and data flow. |
| 1197 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1198 | (blank) | blank separator; no runtime behavior. |
| 1199 |     pub fn per_bus_health_01(&self, bus: BusId) -> f64 { | function signature/body; changes affect API behavio
| 1200 |         let b = self.bus_ref(bus); | support logic; impact depends on neighboring block and data flow. |
| 1201 |         let mut score = 1.0; | support logic; impact depends on neighboring block and data flow. |
| 1202 |         score -= (b.bus_load_pct / 100.0).powf(1.3) * 0.25; | support logic; impact depends on neighboring bloc
```

```
| 1203 |         score -= (b.total_errors as f64 / 2000.0).min(0.35); | support logic; impact depends on neighboring blo
| 1204 |         score -= match b.state { | support logic; impact depends on neighboring block and data flow. |
| 1205 |             BusState::ErrorActive => 0.0, | support logic; impact depends on neighboring block and data flow.
| 1206 |             BusState::ErrorPassive => 0.30, | support logic; impact depends on neighboring block and data flow
| 1207 |             BusState::BusOff => 0.85, | support logic; impact depends on neighboring block and data flow. |
| 1208 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1209 |         score.clamp(0.0, 1.0) | support logic; impact depends on neighboring block and data flow. |
| 1210 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1211 | (blank) | blank separator; no runtime behavior. |
| 1212 |     pub fn network_health_score_01(&self) -> f64 { | function signature/body; changes affect API behavior and c
| 1213 |         let buses = [ | support logic; impact depends on neighboring block and data flow. |
| 1214 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 1215 |             BusId::ChassisHs, | support logic; impact depends on neighboring block and data flow. |
| 1216 |             BusId::BodyMs, | support logic; impact depends on neighboring block and data flow. |
| 1217 |             BusId::IsoBus, | support logic; impact depends on neighboring block and data flow. |
| 1218 |             BusId::Diagnostic, | support logic; impact depends on neighboring block and data flow. |
| 1219 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 1220 |         let avg_bus = buses | support logic; impact depends on neighboring block and data flow. |
| 1221 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1222 |             .map(\|b\| self.per_bus_health_01(*b)) | support logic; impact depends on neighboring block and dat
| 1223 |             .sum::<f64>() | support logic; impact depends on neighboring block and data flow. |
| 1224 |             / buses.len() as f64; | support logic; impact depends on neighboring block and data flow. |
| 1225 |         let online_ratio = if self.nodes.is_empty() { | support logic; impact depends on neighboring block and
| 1226 |             1.0 | support logic; impact depends on neighboring block and data flow. |
| 1227 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 1228 |             self.online_count() as f64 / self.nodes.len() as f64 | support logic; impact depends on neighboring
| 1229 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1230 |         (avg_bus * 0.82 + online_ratio * 0.18).clamp(0.0, 1.0) | support logic; impact depends on neighboring b
| 1231 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1232 | (blank) | blank separator; no runtime behavior. |
| 1233 |     pub fn snapshot(&self) -> CanNetworkSnapshot { | function signature/body; changes affect API behavior and c
| 1234 |         let buses = [ | support logic; impact depends on neighboring block and data flow. |
| 1235 |             (BusId::PowertrainHs, &self.powertrain), | support logic; impact depends on neighboring block and c
| 1236 |             (BusId::ChassisHs, &self.chassis), | support logic; impact depends on neighboring block and data fl
| 1237 |             (BusId::BodyMs, &self.body), | support logic; impact depends on neighboring block and data flow. |
| 1238 |             (BusId::IsoBus, &self.isobus), | support logic; impact depends on neighboring block and data flow.
| 1239 |             (BusId::Diagnostic, &self.diagnostic), | support logic; impact depends on neighboring block and dat
| 1240 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 1241 |         let bus_vec = buses | support logic; impact depends on neighboring block and data flow. |
| 1242 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1243 |             .map(\|(id, b)\| BusSnapshot { | support logic; impact depends on neighboring block and data flow.
| 1244 |                 bus: format!("{}", id), | support logic; impact depends on neighboring block and data flow. |
| 1245 |                 speed_kbps: b.speed.kbps(), | support logic; impact depends on neighboring block and data flow
| 1246 |                 state: format!("{}", b.state), | support logic; impact depends on neighboring block and data fl
| 1247 |                 load_pct: b.bus_load_pct, | support logic; impact depends on neighboring block and data flow.
| 1248 |                 total_tx: b.total_tx, | support logic; impact depends on neighboring block and data flow. |
| 1249 |                 total_errors: b.total_errors, | support logic; impact depends on neighboring block and data flo
| 1250 |                 log_depth: b.frame_log.len(), | support logic; impact depends on neighboring block and data flo
| 1251 |             }) | support logic; impact depends on neighboring block and data flow. |
| 1252 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 1253 | (blank) | blank separator; no runtime behavior. |
| 1254 |         CanNetworkSnapshot { | support logic; impact depends on neighboring block and data flow. |
| 1255 |             elapsed_s: self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 1256 |             health_score_01: self.network_health_score_01(), | support logic; impact depends on neighboring blo
| 1257 |             total_frames_all_buses: self.total_frames_all_buses, | support logic; impact depends on neighboring
| 1258 |             total_errors_all_buses: self.total_errors(), | support logic; impact depends on neighboring block a
| 1259 |             online_nodes: self.online_count(), | support logic; impact depends on neighboring block and data fl
| 1260 |             buses: bus_vec, | support logic; impact depends on neighboring block and data flow. |
| 1261 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1262 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1263 | (blank) | blank separator; no runtime behavior. |
| 1264 |     pub fn export_snapshot_json<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function signature/b
| 1265 |         if let Some(parent) = path.as_ref().parent() { | runtime gate; condition changes can enable/disable cri
| 1266 |             fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation semantics
| 1267 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1268 |         let data = serde_json::to_string_pretty(&self.snapshot()).map_err(\|e\| { | support logic; impact depen
| 1269 |             std::io::Error::other(format!("json serialize: {}", e)) | support logic; impact depends on neighbo
| 1270 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 1271 |         fs::write(path, data) | protocol or I/O critical; changes may impact external integration correctness.
| 1272 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1273 | (blank) | blank separator; no runtime behavior. |
| 1274 |     pub fn export_snapshot_csv<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function signature/bo
| 1275 |         if let Some(parent) = path.as_ref().parent() { | runtime gate; condition changes can enable/disable cri
| 1276 |             fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation semantics
| 1277 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1278 |         let snap = self.snapshot(); | support logic; impact depends on neighboring block and data flow. |
```

```
| 1279 |         let mut f = fs::File::create(path)?; | error propagation point; changes alter failure propagation sema
| 1280 |         writeln!( | protocol or I/O critical; changes may impact external integration correctness. |
| 1281 |             f, | support logic; impact depends on neighboring block and data flow. |
| 1282 |             "elapsed_s,health_score_01,total_frames_all_buses,total_errors_all_buses,online_nodes" | support lo
| 1283 |         )?; | error propagation point; changes alter failure propagation semantics. |
| 1284 |         writeln!( | protocol or I/O critical; changes may impact external integration correctness. |
| 1285 |             f, | support logic; impact depends on neighboring block and data flow. |
| 1286 |             "{:.3},{:.5},{},{},{}", | support logic; impact depends on neighboring block and data flow. |
| 1287 |             snap.elapsed_s, | support logic; impact depends on neighboring block and data flow. |
| 1288 |             snap.health_score_01, | support logic; impact depends on neighboring block and data flow. |
| 1289 |             snap.total_frames_all_buses, | support logic; impact depends on neighboring block and data flow. |
| 1290 |             snap.total_errors_all_buses, | support logic; impact depends on neighboring block and data flow. |
| 1291 |             snap.online_nodes | support logic; impact depends on neighboring block and data flow. |
| 1292 |         )?; | error propagation point; changes alter failure propagation semantics. |
| 1293 |         writeln!(f)?; | error propagation point; changes alter failure propagation semantics. |
| 1294 |         writeln!( | protocol or I/O critical; changes may impact external integration correctness. |
| 1295 |             f, | support logic; impact depends on neighboring block and data flow. |
| 1296 |             "bus,speed_kbps,state,load_pct,total_tx,total_errors,log_depth" | support logic; impact depends on
| 1297 |         )?; | error propagation point; changes alter failure propagation semantics. |
| 1298 |         for b in snap.buses { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1299 |             writeln!( | protocol or I/O critical; changes may impact external integration correctness. |
| 1300 |                 f, | support logic; impact depends on neighboring block and data flow. |
| 1301 |                 "\"{}\",{},\"{}\",{:.4},{},{},{}", | support logic; impact depends on neighboring block and dat
| 1302 |                 b.bus, b.speed_kbps, b.state, b.load_pct, b.total_tx, b.total_errors, b.log_depth | support log
| 1303 |             )?; | error propagation point; changes alter failure propagation semantics. |
| 1304 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1305 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 1306 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1307 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1308 | (blank) | blank separator; no runtime behavior. |
| 1309 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 1310 | mod tests { | module declaration; changing can break namespace graph. |
| 1311 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 1312 | (blank) | blank separator; no runtime behavior. |
| 1313 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1314 |     fn arbitration_orders_by_can_id() { | function signature/body; changes affect API behavior and control-flow
| 1315 |         let mut bus = CanBus::new(BusId::PowertrainHs, CanSpeed::Kbps500); | support logic; impact depends on
| 1316 |         let mut nodes: HashMap<u8, CanNode> = HashMap::new(); | support logic; impact depends on neighboring b
| 1317 |         nodes.insert( | support logic; impact depends on neighboring block and data flow. |
| 1318 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 1319 |             CanNode::new(addr::ECM_1, "ECM", BusId::PowertrainHs), | support logic; impact depends on neighbori
| 1320 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1321 |         nodes.insert( | support logic; impact depends on neighboring block and data flow. |
| 1322 |             addr::TRANSMISSION, | support logic; impact depends on neighboring block and data flow. |
| 1323 |             CanNode::new(addr::TRANSMISSION, "TCM", BusId::PowertrainHs), | support logic; impact depends on ne
| 1324 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1325 | (blank) | blank separator; no runtime behavior. |
| 1326 |         let id_low = J1939Frame::build_id(2, pgn::EBC1, addr::TRANSMISSION, 0xFF); | support logic; impact depe
| 1327 |         let id_high = J1939Frame::build_id(6, pgn::DM1, addr::ECM_1, 0xFF); | support logic; impact depends on
| 1328 |         bus.transmit(J1939Frame::from_raw(1.0, id_high, &[0; 8])); | support logic; impact depends on neighbori
| 1329 |         bus.transmit(J1939Frame::from_raw(1.0, id_low, &[0; 8])); | support logic; impact depends on neighborin
| 1330 |         bus.tick(&mut nodes, 0.01); | support logic; impact depends on neighboring block and data flow. |
| 1331 | (blank) | blank separator; no runtime behavior. |
| 1332 |         assert_eq!(bus.dispatched.len(), 2); | support logic; impact depends on neighboring block and data flow
| 1333 |         assert!(bus.dispatched[0].raw_id < bus.dispatched[1].raw_id); | support logic; impact depends on neighb
| 1334 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1335 | (blank) | blank separator; no runtime behavior. |
| 1336 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1337 |     fn scheduler_jitter_stays_in_expected_band() { | function signature/body; changes affect API behavior and c
| 1338 |         let mut s = MessageSchedule::new( | support logic; impact depends on neighboring block and data flow.
| 1339 |             pgn::EEC1, | support logic; impact depends on neighboring block and data flow. |
| 1340 |             addr::ECM_1, | support logic; impact depends on neighboring block and data flow. |
| 1341 |             0xFF, | support logic; impact depends on neighboring block and data flow. |
| 1342 |             BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow. |
| 1343 |             100.0, | support logic; impact depends on neighboring block and data flow. |
| 1344 |             3, | support logic; impact depends on neighboring block and data flow. |
| 1345 |             "test", | support logic; impact depends on neighboring block and data flow. |
| 1346 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1347 |         s.jitter_ms = 4.0; | support logic; impact depends on neighboring block and data flow. |
| 1348 |         let mut t_ms = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1349 |         let mut fire_times: Vec<f64> = Vec::new(); | support logic; impact depends on neighboring block and dat
| 1350 |         while fire_times.len() < 12 { | iteration logic; may alter timing, throughput, and safety constraints.
| 1351 |             t_ms += 1.0; | support logic; impact depends on neighboring block and data flow. |
| 1352 |             if s.should_fire(1.0, 0.5) { | runtime gate; condition changes can enable/disable critical paths.
| 1353 |                 fire_times.push(t_ms); | support logic; impact depends on neighboring block and data flow. |
| 1354 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 1355 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1356 |         for w in fire_times.windows(2) { | iteration logic; may alter timing, throughput, and safety constraint
| 1357 |             let dt = w[1] - w[0]; | support logic; impact depends on neighboring block and data flow. |
| 1358 |             assert!((98.0..=104.5).contains(&dt), "interval {} out of band", dt); | support logic; impact depen
| 1359 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1360 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1361 | (blank) | blank separator; no runtime behavior. |
| 1362 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1363 |     fn routed_message_appears_next_tick_on_destination_bus() { | function signature/body; changes affect API be
| 1364 |         let mut net = CanNetwork::new(); | support logic; impact depends on neighboring block and data flow. |
| 1365 |         net.set_online(addr::ECM_1); | support logic; impact depends on neighboring block and data flow. |
| 1366 |         net.set_online(addr::INSTRUMENT); | support logic; impact depends on neighboring block and data flow.
| 1367 | (blank) | blank separator; no runtime behavior. |
| 1368 |         let id = J1939Frame::build_id(3, pgn::EEC1, addr::ECM_1, 0xFF); | support logic; impact depends on neig
| 1369 |         let frame = J1939Frame::from_raw(0.0, id, &[0xFF; 8]); | support logic; impact depends on neighboring b
| 1370 |         net.transmit(frame); | support logic; impact depends on neighboring block and data flow. |
| 1371 |         net.tick(0.01); | support logic; impact depends on neighboring block and data flow. |
| 1372 | (blank) | blank separator; no runtime behavior. |
| 1373 |         let body_after_first = net.body.total_tx; | support logic; impact depends on neighboring block and data
| 1374 |         net.tick(0.01); | support logic; impact depends on neighboring block and data flow. |
| 1375 |         let body_after_second = net.body.total_tx; | support logic; impact depends on neighboring block and dat
| 1376 |         assert!( | support logic; impact depends on neighboring block and data flow. |
| 1377 |             body_after_second > body_after_first, | support logic; impact depends on neighboring block and data
| 1378 |             "expected routed frame on next tick" | support logic; impact depends on neighboring block and data
| 1379 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1380 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1381 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/chassis.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 184 | Non-empty: 164 | Symbol count: 8

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 2 |
| update | 1 |
| reset | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | const | WHEELBASE | namespace and compile-time linkage |
| 4 | enum | WheelState | data/schema coupling and match/serde break risk |
| 12 | enum | EpsMode | data/schema coupling and match/serde break risk |
| 20 | struct | ChassisControl | data/schema coupling and match/serde break risk |
| 46 | impl | ChassisControl | method behavior and trait semantics |
| 47 | fn | new | API and behavior coupling to callers/implementers |
| 67 | fn | update | API and behavior coupling to callers/implementers |
| 181 | fn | reset | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | const WHEELBASE: f64 = 2.7; // m | support logic; impact depends on neighboring block and data flow. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent tracea
| 4 | pub enum WheelState { | state machine variants; changes ripple through matches and business logic. |
| 5 |     Rolling, | support logic; impact depends on neighboring block and data flow. |
| 6 |     Skidding, | support logic; impact depends on neighboring block and data flow. |
| 7 |     AbsActive, | support logic; impact depends on neighboring block and data flow. |
| 8 |     TcsLimited, | support logic; impact depends on neighboring block and data flow. |
| 9 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 10 | (blank) | blank separator; no runtime behavior. |
| 11 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 12 | pub enum EpsMode { | state machine variants; changes ripple through matches and business logic. |
| 13 |     Normal, | support logic; impact depends on neighboring block and data flow. |
| 14 |     Sport, | support logic; impact depends on neighboring block and data flow. |
```

```
| 15 |     Comfort, | support logic; impact depends on neighboring block and data flow. |
| 16 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | /// Chassis domain controller: ABS + ESP + TCS + EPS | comment/documentation; changing affects understanding and
| 19 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 20 | pub struct ChassisControl { | data layout contract; field changes can break callers/serde/tests. |
| 21 |     // ABS | comment/documentation; changing affects understanding and intent traceability. |
| 22 |     pub abs_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 23 |     pub abs_cycles: u32, | support logic; impact depends on neighboring block and data flow. |
| 24 |     abs_cycle_phases: [f64; 4], | support logic; impact depends on neighboring block and data flow. |
| 25 | (blank) | blank separator; no runtime behavior. |
| 26 |     // ESP — Electronic Stability Program | comment/documentation; changing affects understanding and intent trac
| 27 |     pub esp_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 28 |     pub oversteer: bool, | support logic; impact depends on neighboring block and data flow. |
| 29 |     pub understeer: bool, | support logic; impact depends on neighboring block and data flow. |
| 30 |     pub esp_brake_correction: [f64; 4], // individual wheel brake delta | support logic; impact depends on neighb
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 |     // TCS — Traction Control | comment/documentation; changing affects understanding and intent traceability. |
| 33 |     pub tcs_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 34 |     pub tcs_throttle_cut: f64, // 0-1 fraction cut | support logic; impact depends on neighboring block and data
| 35 | (blank) | blank separator; no runtime behavior. |
| 36 |     // EPS — Electronic Power Steering | comment/documentation; changing affects understanding and intent traceab
| 37 |     pub eps_assist_torque: f64, // Nm | support logic; impact depends on neighboring block and data flow. |
| 38 |     pub eps_mode: EpsMode, | support logic; impact depends on neighboring block and data flow. |
| 39 | (blank) | blank separator; no runtime behavior. |
| 40 |     // Per-wheel state | comment/documentation; changing affects understanding and intent traceability. |
| 41 |     pub wheel_states: [WheelState; 4], | support logic; impact depends on neighboring block and data flow. |
| 42 |     pub brake_pressures: [f64; 4], | support logic; impact depends on neighboring block and data flow. |
| 43 |     pub wheel_velocities: [f64; 4], | support logic; impact depends on neighboring block and data flow. |
| 44 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 45 | (blank) | blank separator; no runtime behavior. |
| 46 | impl ChassisControl { | implementation binding; behavior semantics and trait conformance live here. |
| 47 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 48 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 49 |             abs_active: false, | support logic; impact depends on neighboring block and data flow. |
| 50 |             abs_cycles: 0, | support logic; impact depends on neighboring block and data flow. |
| 51 |             abs_cycle_phases: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 52 |             esp_active: false, | support logic; impact depends on neighboring block and data flow. |
| 53 |             oversteer: false, | support logic; impact depends on neighboring block and data flow. |
| 54 |             understeer: false, | support logic; impact depends on neighboring block and data flow. |
| 55 |             esp_brake_correction: [0.0; 4], | support logic; impact depends on neighboring block and data flow.
| 56 |             tcs_active: false, | support logic; impact depends on neighboring block and data flow. |
| 57 |             tcs_throttle_cut: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 58 |             eps_assist_torque: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 59 |             eps_mode: EpsMode::Normal, | support logic; impact depends on neighboring block and data flow. |
| 60 |             wheel_states: [WheelState::Rolling; 4], | support logic; impact depends on neighboring block and data
| 61 |             brake_pressures: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 62 |             wheel_velocities: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 63 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 64 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 65 | (blank) | blank separator; no runtime behavior. |
| 66 |     /// Returns effective throttle after TCS cut | comment/documentation; changing affects understanding and inte
| 67 |     pub fn update( | function signature/body; changes affect API behavior and control-flow. |
| 68 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 69 |         vehicle_speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 70 |         steering_angle: f64, | support logic; impact depends on neighboring block and data flow. |
| 71 |         yaw_rate: f64, | support logic; impact depends on neighboring block and data flow. |
| 72 |         user_brake: f64, | support logic; impact depends on neighboring block and data flow. |
| 73 |         user_throttle: f64, | support logic; impact depends on neighboring block and data flow. |
| 74 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 75 |     ) -> (f64, f64) { | support logic; impact depends on neighboring block and data flow. |
| 76 |         // --- ABS ---------------------------------------------------------- | comment/documentation; changing a
| 77 |         let mut effective_brake = user_brake; | support logic; impact depends on neighboring block and data flow
| 78 |         self.abs_active = false; | support logic; impact depends on neighboring block and data flow. |
| 79 | (blank) | blank separator; no runtime behavior. |
| 80 |         for i in 0..4 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 81 |             // Wheel dynamics: brake slows wheel faster than vehicle | comment/documentation; changing affects u
| 82 |             let decel = self.brake_pressures[i] * 12.0; | support logic; impact depends on neighboring block and
| 83 |             self.wheel_velocities[i] = (self.wheel_velocities[i] - decel * dt).max(0.0); | support logic; impact
| 84 |             // Follow vehicle speed when not braking | comment/documentation; changing affects understanding and
| 85 |             self.wheel_velocities[i] += | support logic; impact depends on neighboring block and data flow. |
| 86 |                 ((vehicle_speed - self.wheel_velocities[i]) * 0.3 * dt).max(0.0); | support logic; impact depends
| 87 | (blank) | blank separator; no runtime behavior. |
| 88 |             let slip = vehicle_speed - self.wheel_velocities[i]; | support logic; impact depends on neighboring b
| 89 |             if slip > 5.0 && user_brake > 0.3 && vehicle_speed > 5.0 { | runtime gate; condition changes can enab
| 90 |                 self.wheel_states[i] = WheelState::Skidding; | support logic; impact depends on neighboring block
```

```
| 91 |                 self.abs_active = true; | support logic; impact depends on neighboring block and data flow. |
| 92 |                 self.abs_cycles += 1; | support logic; impact depends on neighboring block and data flow. |
| 93 |             } else if user_brake > 0.05 { | support logic; impact depends on neighboring block and data flow. |
| 94 |                 self.wheel_states[i] = WheelState::AbsActive; | support logic; impact depends on neighboring bloc
| 95 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 96 |                 self.wheel_states[i] = WheelState::Rolling; | support logic; impact depends on neighboring block
| 97 |                 self.wheel_velocities[i] = vehicle_speed; // sync | support logic; impact depends on neighboring
| 98 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 99 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 100 | (blank) | blank separator; no runtime behavior. |
| 101 |         // ABS modulates brake pressure at 8 Hz | comment/documentation; changing affects understanding and inte
| 102 |         if self.abs_active { | runtime gate; condition changes can enable/disable critical paths. |
| 103 |             for i in 0..4 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 104 |                 self.abs_cycle_phases[i] += dt * 8.0; | support logic; impact depends on neighboring block and
| 105 |                 if self.abs_cycle_phases[i] > 1.0 { | runtime gate; condition changes can enable/disable critica
| 106 |                     self.abs_cycle_phases[i] -= 1.0; | support logic; impact depends on neighboring block and da
| 107 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 108 |                 let modulated = if self.abs_cycle_phases[i] < 0.45 { | support logic; impact depends on neighbo
| 109 |                     user_brake * 0.28 | support logic; impact depends on neighboring block and data flow. |
| 110 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 111 |                     user_brake * 0.92 | support logic; impact depends on neighboring block and data flow. |
| 112 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 113 |                 self.brake_pressures[i] = modulated; | support logic; impact depends on neighboring block and da
| 114 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 115 |             effective_brake = self.brake_pressures.iter().sum::<f64>() / 4.0; | support logic; impact depends o
| 116 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 117 |             for p in &mut self.brake_pressures { | iteration logic; may alter timing, throughput, and safety co
| 118 |                 *p = user_brake; | support logic; impact depends on neighboring block and data flow. |
| 119 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 120 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 121 | (blank) | blank separator; no runtime behavior. |
| 122 |         // --- TCS ---------------------------------------------------------- | comment/documentation; changing
| 123 |         let mut effective_throttle = user_throttle; | support logic; impact depends on neighboring block and da
| 124 |         self.tcs_active = false; | support logic; impact depends on neighboring block and data flow. |
| 125 |         self.tcs_throttle_cut = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 126 | (blank) | blank separator; no runtime behavior. |
| 127 |         if user_throttle > 0.2 && vehicle_speed < 60.0 { | runtime gate; condition changes can enable/disable c
| 128 |             // Detect wheel spin (driven wheels RL/RR faster than vehicle) | comment/documentation; changing af
| 129 |             let driven_avg = (self.wheel_velocities[2] + self.wheel_velocities[3]) / 2.0; | support logic; impac
| 130 |             if driven_avg > vehicle_speed + 6.0 { | runtime gate; condition changes can enable/disable critical
| 131 |                 self.tcs_active = true; | support logic; impact depends on neighboring block and data flow. |
| 132 |                 self.tcs_throttle_cut = ((driven_avg - vehicle_speed - 6.0) / 10.0).min(0.8); | support logic; i
| 133 |                 effective_throttle *= 1.0 - self.tcs_throttle_cut; | support logic; impact depends on neighborin
| 134 |                 // Mark driven wheels | comment/documentation; changing affects understanding and intent traceab
| 135 |                 self.wheel_states[2] = WheelState::TcsLimited; | support logic; impact depends on neighboring b
| 136 |                 self.wheel_states[3] = WheelState::TcsLimited; | support logic; impact depends on neighboring b
| 137 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 139 | (blank) | blank separator; no runtime behavior. |
| 140 |         // --- ESP ---------------------------------------------------------- | comment/documentation; changing
| 141 |         self.esp_active = false; | support logic; impact depends on neighboring block and data flow. |
| 142 |         self.esp_brake_correction = [0.0; 4]; | support logic; impact depends on neighboring block and data flo
| 143 | (blank) | blank separator; no runtime behavior. |
| 144 |         if vehicle_speed > 20.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 145 |             // Theoretical yaw rate for neutral steer | comment/documentation; changing affects understanding a
| 146 |             let yaw_neutral = (vehicle_speed / 3.6) * (steering_angle.to_radians()) / WHEELBASE; | support logic
| 147 |             let yaw_error = yaw_rate.to_radians() - yaw_neutral; | support logic; impact depends on neighboring
| 148 | (blank) | blank separator; no runtime behavior. |
| 149 |             if yaw_error.abs() > 0.08 { | runtime gate; condition changes can enable/disable critical paths. |
| 150 |                 self.esp_active = true; | support logic; impact depends on neighboring block and data flow. |
| 151 |                 if yaw_error > 0.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 152 |                     // Oversteer: apply outer front wheel brake | comment/documentation; changing affects unders
| 153 |                     self.oversteer = true; | support logic; impact depends on neighboring block and data flow.
| 154 |                     self.understeer = false; | support logic; impact depends on neighboring block and data flow
| 155 |                     self.esp_brake_correction[1] = (yaw_error * 3.0).min(0.6); // FR | support logic; impact de
| 156 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 157 |                     // Understeer: apply inner rear wheel brake | comment/documentation; changing affects unders
| 158 |                     self.understeer = true; | support logic; impact depends on neighboring block and data flow.
| 159 |                     self.oversteer = false; | support logic; impact depends on neighboring block and data flow.
| 160 |                     self.esp_brake_correction[2] = ((-yaw_error) * 2.0).min(0.5); | support logic; impact depen
| 161 |                     // RL | comment/documentation; changing affects understanding and intent traceability. |
| 162 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 163 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 164 |                 self.oversteer = false; | support logic; impact depends on neighboring block and data flow. |
| 165 |                 self.understeer = false; | support logic; impact depends on neighboring block and data flow. |
| 166 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 167 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 168 | (blank) | blank separator; no runtime behavior. |
| 169 |         // --- EPS ------------------------------------------------------- | comment/documentation; changing
| 170 |         // Assist drops at high speed (less assist needed) | comment/documentation; changing affects understandi
| 171 |         let speed_factor = (1.0 - vehicle_speed / 200.0).clamp(0.2, 1.0); | support logic; impact depends on nei
| 172 |         self.eps_assist_torque = match self.eps_mode { | support logic; impact depends on neighboring block and
| 173 |             EpsMode::Comfort => steering_angle.abs() * 0.12 * speed_factor, | support logic; impact depends on n
| 174 |             EpsMode::Normal => steering_angle.abs() * 0.08 * speed_factor, | support logic; impact depends on ne
| 175 |             EpsMode::Sport => steering_angle.abs() * 0.04 * speed_factor, | support logic; impact depends on nei
| 176 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 177 | (blank) | blank separator; no runtime behavior. |
| 178 |         (effective_throttle, effective_brake) | support logic; impact depends on neighboring block and data flow
| 179 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 180 | (blank) | blank separator; no runtime behavior. |
| 181 |     pub fn reset(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 182 |         *self = Self::new(); | support logic; impact depends on neighboring block and data flow. |
| 183 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 184 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/ecm_heavy.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 376 | Non-empty: 341 | Symbol count: 23

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| inject_dtc | 6 |
| new | 4 |
| fmt | 2 |
| torque_curve | 2 |
| check_faults | 2 |
| default | 1 |
| update | 1 |
| push | 1 |
| clear_dtcs | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 7 | const | IDLE_RPM | namespace and compile-time linkage |
| 8 | const | RATED_RPM | namespace and compile-time linkage |
| 9 | const | MAX_RPM | namespace and compile-time linkage |
| 11 | const | REDLINE_RPM | namespace and compile-time linkage |
| 12 | const | PEAK_TORQUE_NM | namespace and compile-time linkage |
| 13 | const | FUEL_TANK_L | namespace and compile-time linkage |
| 14 | const | NUM_CYLINDERS | namespace and compile-time linkage |
| 18 | enum | AftertreatmentState | data/schema coupling and match/serde break risk |
| 25 | impl | std::fmt::Display for AftertreatmentState | method behavior and trait semantics |
| 26 | fn | fmt | API and behavior coupling to callers/implementers |
| 38 | enum | GovernorMode | data/schema coupling and match/serde break risk |
| 45 | impl | std::fmt::Display for GovernorMode | method behavior and trait semantics |
| 46 | fn | fmt | API and behavior coupling to callers/implementers |
| 58 | struct | HeavyEcm | data/schema coupling and match/serde break risk |
| 131 | impl | Default for HeavyEcm | method behavior and trait semantics |
| 132 | fn | default | API and behavior coupling to callers/implementers |
| 137 | impl | HeavyEcm | method behavior and trait semantics |
| 138 | fn | new | API and behavior coupling to callers/implementers |
| 193 | fn | torque_curve | API and behavior coupling to callers/implementers |
| 209 | fn | update | API and behavior coupling to callers/implementers |
| 292 | fn | check_faults | API and behavior coupling to callers/implementers |
| 353 | fn | inject_dtc | API and behavior coupling to callers/implementers |
| 370 | fn | clear_dtcs | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Heavy Machinery Engine Control Module (ECM) | comment/documentation; changing affects understanding and intent
| 2 | //! Models a Tier 4 / Stage V diesel engine (6-cylinder, ~200 kW) | comment/documentation; changing affects unders
```

```
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use crate::j1939::{Dtc, DtcSeverity}; | import wiring; changing path/name can break compilation and module linkage
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | // Engine configuration constants | comment/documentation; changing affects understanding and intent traceability
| 7 | const IDLE_RPM: f64 = 800.0; | support logic; impact depends on neighboring block and data flow. |
| 8 | const RATED_RPM: f64 = 2200.0; | support logic; impact depends on neighboring block and data flow. |
| 9 | const MAX_RPM: f64 = 2600.0; | support logic; impact depends on neighboring block and data flow. |
| 10 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 11 | const REDLINE_RPM: f64 = 2400.0; | support logic; impact depends on neighboring block and data flow. |
| 12 | const PEAK_TORQUE_NM: f64 = 1000.0; // ~900 Nm at 1400 RPM | support logic; impact depends on neighboring block a
| 13 | const FUEL_TANK_L: f64 = 200.0; | support logic; impact depends on neighboring block and data flow. |
| 14 | const NUM_CYLINDERS: u8 = 6; | support logic; impact depends on neighboring block and data flow. |
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | /// Aftertreatment / DEF / DPF state | comment/documentation; changing affects understanding and intent traceabi
| 17 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 18 | pub enum AftertreatmentState { | state machine variants; changes ripple through matches and business logic. |
| 19 |     Normal, | support logic; impact depends on neighboring block and data flow. |
| 20 |     Regenerating, | support logic; impact depends on neighboring block and data flow. |
| 21 |     HighSoot, | support logic; impact depends on neighboring block and data flow. |
| 22 |     Fault, | support logic; impact depends on neighboring block and data flow. |
| 23 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 | impl std::fmt::Display for AftertreatmentState { | implementation binding; behavior semantics and trait conformar
| 26 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 27 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 28 |             AftertreatmentState::Normal => write!(f, "NORMAL    "), | protocol or I/O critical; changes may impac
| 29 |             AftertreatmentState::Regenerating => write!(f, "REGEN     "), | protocol or I/O critical; changes may
| 30 |             AftertreatmentState::HighSoot => write!(f, "HIGH SOOT "), | protocol or I/O critical; changes may im
| 31 |             AftertreatmentState::Fault => write!(f, "FAULT     "), | protocol or I/O critical; changes may impact
| 32 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 35 | (blank) | blank separator; no runtime behavior. |
| 36 | /// Governor mode | comment/documentation; changing affects understanding and intent traceability. |
| 37 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 38 | pub enum GovernorMode { | state machine variants; changes ripple through matches and business logic. |
| 39 |     Low, | support logic; impact depends on neighboring block and data flow. |
| 40 |     High, | support logic; impact depends on neighboring block and data flow. |
| 41 |     Droop, | support logic; impact depends on neighboring block and data flow. |
| 42 |     All, | support logic; impact depends on neighboring block and data flow. |
| 43 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 44 | (blank) | blank separator; no runtime behavior. |
| 45 | impl std::fmt::Display for GovernorMode { | implementation binding; behavior semantics and trait conformance live
| 46 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 47 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 48 |             GovernorMode::Low => write!(f, "LOW IDLE"), | protocol or I/O critical; changes may impact external i
| 49 |             GovernorMode::High => write!(f, "HIGH    "), | protocol or I/O critical; changes may impact external
| 50 |             GovernorMode::Droop => write!(f, "DROOP   "), | protocol or I/O critical; changes may impact externa
| 51 |             GovernorMode::All => write!(f, "ALL SPD "), | protocol or I/O critical; changes may impact external i
| 52 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 53 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 | (blank) | blank separator; no runtime behavior. |
| 56 | /// Comprehensive Heavy Machinery ECM | comment/documentation; changing affects understanding and intent traceabi
| 57 | #[derive(Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 58 | pub struct HeavyEcm { | data layout contract; field changes can break callers/serde/tests. |
| 59 |     // ■ Engine Speed ■■■
```

```
| 79 |     pub fuel_pressure_kpa: f64, | support logic; impact depends on neighboring block and data flow. |
| 80 |     pub fuel_rate_lph: f64, | support logic; impact depends on neighboring block and data flow. |
| 81 |     pub total_fuel_l: f64, | support logic; impact depends on neighboring block and data flow. |
| 82 |     pub trip_fuel_l: f64, | support logic; impact depends on neighboring block and data flow. |
| 83 | (blank) | blank separator; no runtime behavior. |
| 84 |     // ■ Temperatures ■■■
```

```
| 155 |             total_fuel_l: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 156 |             trip_fuel_l: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 157 |             coolant_temp_c: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 158 |             oil_temp_c: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 159 |             fuel_temp_c: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 160 |             intake_temp_c: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 161 |             exhaust_temp_c: 200.0, | support logic; impact depends on neighboring block and data flow. |
| 162 |             turbo_oil_temp_c: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 163 |             oil_pressure_kpa: 350.0, | support logic; impact depends on neighboring block and data flow. |
| 164 |             oil_level_pct: 95.0, | support logic; impact depends on neighboring block and data flow. |
| 165 |             coolant_pres_kpa: 50.0, | support logic; impact depends on neighboring block and data flow. |
| 166 |             boost_pressure_kpa: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 167 |             intake_manifold_kpa: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 168 |             air_filter_dp_kpa: 1.5, | support logic; impact depends on neighboring block and data flow. |
| 169 |             battery_v: 12.8, | support logic; impact depends on neighboring block and data flow. |
| 170 |             alternator_v: 14.2, | support logic; impact depends on neighboring block and data flow. |
| 171 |             starter_active: false, | support logic; impact depends on neighboring block and data flow. |
| 172 |             def_level_pct: 72.0, | support logic; impact depends on neighboring block and data flow. |
| 173 |             def_quality_pct: 98.5, | support logic; impact depends on neighboring block and data flow. |
| 174 |             dpf_soot_pct: 15.0, | support logic; impact depends on neighboring block and data flow. |
| 175 |             dpf_temp_c: 250.0, | support logic; impact depends on neighboring block and data flow. |
| 176 |             scr_efficiency_pct: 96.0, | support logic; impact depends on neighboring block and data flow. |
| 177 |             nox_ppm: 45.0, | support logic; impact depends on neighboring block and data flow. |
| 178 |             aftertreatment: AftertreatmentState::Normal, | support logic; impact depends on neighboring block an
| 179 |             engine_hours: 2347.5, | support logic; impact depends on neighboring block and data flow. |
| 180 |             service_hours_left: 152.5, | support logic; impact depends on neighboring block and data flow. |
| 181 |             num_cylinders: NUM_CYLINDERS, | support logic; impact depends on neighboring block and data flow. |
| 182 |             active_dtcs: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 183 |             stored_dtcs: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 184 |             mil_active: false, | support logic; impact depends on neighboring block and data flow. |
| 185 |             amber_lamp: false, | support logic; impact depends on neighboring block and data flow. |
| 186 |             red_lamp: false, | support logic; impact depends on neighboring block and data flow. |
| 187 |             rpm_integral: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 188 |             warmup_complete: false, | support logic; impact depends on neighboring block and data flow. |
| 189 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 190 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 191 | (blank) | blank separator; no runtime behavior. |
| 192 |     /// Diesel torque curve: peak ~1000 Nm at 1200-1600 RPM | comment/documentation; changing affects understand
| 193 |     pub fn torque_curve(&self, rpm: f64, throttle_pct: f64) -> f64 { | function signature/body; changes affect A
| 194 |         let t = throttle_pct / 100.0; | support logic; impact depends on neighboring block and data flow. |
| 195 |         let peak_rpm = 1400.0; | support logic; impact depends on neighboring block and data flow. |
| 196 |         let _norm = (rpm - peak_rpm) / 800.0; | support logic; impact depends on neighboring block and data flo
| 197 |         let curve = if rpm < 800.0 { | support logic; impact depends on neighboring block and data flow. |
| 198 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 199 |         } else if rpm < peak_rpm { | support logic; impact depends on neighboring block and data flow. |
| 200 |             0.7 + 0.3 * (rpm - 800.0) / (peak_rpm - 800.0) | support logic; impact depends on neighboring block
| 201 |         } else if rpm < RATED_RPM { | support logic; impact depends on neighboring block and data flow. |
| 202 |             1.0 - 0.25 * (rpm - peak_rpm) / (RATED_RPM - peak_rpm) | support logic; impact depends on neighbori
| 203 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 204 |             (0.75 - (rpm - RATED_RPM) / (MAX_RPM - RATED_RPM) * 0.75).max(0.0) | support logic; impact depends o
| 205 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 206 |         PEAK_TORQUE_NM * curve * t | support logic; impact depends on neighboring block and data flow. |
| 207 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 208 | (blank) | blank separator; no runtime behavior. |
| 209 |     pub fn update(&mut self, throttle_pct: f64, load_nm: f64, dt: f64) { | function signature/body; changes affe
| 210 |         self.throttle_pct = throttle_pct.clamp(0.0, 100.0); | support logic; impact depends on neighboring bloc
| 211 |         let t = self.throttle_pct / 100.0; | support logic; impact depends on neighboring block and data flow.
| 212 | (blank) | blank separator; no runtime behavior. |
| 213 |         // RPM dynamics: governor tries to maintain rated RPM under load | comment/documentation; changing affe
| 214 |         let target_rpm = self.idle_rpm + t * (RATED_RPM - self.idle_rpm); | support logic; impact depends on ne
| 215 |         let available_torque = self.torque_curve(self.rpm, self.throttle_pct); | support logic; impact depends o
| 216 |         let net_torque = available_torque - load_nm; | support logic; impact depends on neighboring block and da
| 217 |         // d(rpm)/dt based on engine inertia | comment/documentation; changing affects understanding and intent
| 218 |         let rpm_accel = net_torque * 30.0 / (std::f64::consts::PI * 2.5); // J=2.5 kg.m² | support logic; impac
| 219 |         self.rpm_integral += rpm_accel * dt; | support logic; impact depends on neighboring block and data flow
| 220 |         self.rpm += (target_rpm - self.rpm) * dt * 1.5 + rpm_accel * dt * 0.1; | support logic; impact depends o
| 221 |         self.rpm = self.rpm.clamp(self.idle_rpm * 0.8, MAX_RPM); | support logic; impact depends on neighboring
| 222 | (blank) | blank separator; no runtime behavior. |
| 223 |         // Torque & load | comment/documentation; changing affects understanding and intent traceability. |
| 224 |         self.torque_nm = available_torque.min(load_nm + 50.0); | support logic; impact depends on neighboring b
| 225 |         self.actual_torque_pct = (self.torque_nm / PEAK_TORQUE_NM * 100.0).clamp(-100.0, 100.0); | support logic
| 226 |         self.demand_torque_pct = (t * 100.0 - 125.0 + 125.0).clamp(-100.0, 100.0); | support logic; impact depe
| 227 |         self.percent_load = (load_nm / PEAK_TORQUE_NM * 100.0).clamp(0.0, 150.0); | support logic; impact depen
| 228 | (blank) | blank separator; no runtime behavior. |
| 229 |         // Fuel consumption (diesel: ~0.22 kg/kWh specific consumption) | comment/documentation; changing affec
| 230 |         let power_kw = self.torque_nm * self.rpm * std::f64::consts::PI / 30000.0; | support logic; impact depe
```

```
| 231 |         self.fuel_rate_lph = (power_kw * 0.22 / 0.85 + 1.5).max(1.5); // 0.85 kg/L diesel | support logic; impac
| 232 |         let consumed = self.fuel_rate_lph * dt / 3600.0; | support logic; impact depends on neighboring block a
| 233 |         self.total_fuel_l += consumed; | support logic; impact depends on neighboring block and data flow. |
| 234 |         self.trip_fuel_l += consumed; | support logic; impact depends on neighboring block and data flow. |
| 235 |         self.fuel_level_pct = (self.fuel_level_pct - consumed / FUEL_TANK_L * 100.0).max(0.0); | support logic;
| 236 | (blank) | blank separator; no runtime behavior. |
| 237 |         // Thermal model | comment/documentation; changing affects understanding and intent traceability. |
| 238 |         if !self.warmup_complete { | runtime gate; condition changes can enable/disable critical paths. |
| 239 |             self.warmup_complete = self.coolant_temp_c > 80.0; | support logic; impact depends on neighboring b
| 240 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 241 |         let tgt_coolant = if self.warmup_complete { | support logic; impact depends on neighboring block and da
| 242 |             90.0 + t * 5.0 | support logic; impact depends on neighboring block and data flow. |
| 243 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 244 |             self.coolant_temp_c + 3.0 * dt | support logic; impact depends on neighboring block and data flow.
| 245 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 246 |         self.coolant_temp_c += (tgt_coolant - self.coolant_temp_c) * dt * 0.02; | support logic; impact depends
| 247 |         self.oil_temp_c = self.coolant_temp_c + 12.0; | support logic; impact depends on neighboring block and
| 248 |         self.exhaust_temp_c = 200.0 + t * 450.0 * (self.rpm / RATED_RPM).min(1.0); | support logic; impact depe
| 249 |         self.intake_temp_c = 25.0 + self.boost_pressure_kpa / 100.0 * 15.0; | support logic; impact depends on
| 250 | (blank) | blank separator; no runtime behavior. |
| 251 |         // Pressures | comment/documentation; changing affects understanding and intent traceability. |
| 252 |         let rpm_factor = (self.rpm / RATED_RPM).clamp(0.2, 1.2); | support logic; impact depends on neighboring
| 253 |         self.oil_pressure_kpa = 250.0 + rpm_factor * 200.0; | support logic; impact depends on neighboring bloc
| 254 |         self.boost_pressure_kpa = 100.0 + t * rpm_factor * 200.0; | support logic; impact depends on neighborin
| 255 |         self.intake_manifold_kpa = 100.0 + self.boost_pressure_kpa * 0.8; | support logic; impact depends on ne
| 256 |         self.fuel_pressure_kpa = 350.0 + t * 100.0; | support logic; impact depends on neighboring block and da
| 257 | (blank) | blank separator; no runtime behavior. |
| 258 |         // Aftertreatment | comment/documentation; changing affects understanding and intent traceability. |
| 259 |         self.dpf_soot_pct = (self.dpf_soot_pct + t * 0.0005 * dt - 0.0001 * dt).clamp(0.0, 100.0); | support lo
| 260 |         self.dpf_temp_c = 250.0 + t * 300.0; | support logic; impact depends on neighboring block and data flow
| 261 |         self.def_level_pct = | support logic; impact depends on neighboring block and data flow. |
| 262 |             (self.def_level_pct - self.fuel_rate_lph * 0.05 * dt / 3600.0 * 100.0).max(0.0); | support logic; i
| 263 |         self.scr_efficiency_pct = if self.def_level_pct > 5.0 { | support logic; impact depends on neighboring
| 264 |             94.0 + t * 3.0 | support logic; impact depends on neighboring block and data flow. |
| 265 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 266 |             20.0 | support logic; impact depends on neighboring block and data flow. |
| 267 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 268 |         self.nox_ppm = (1500.0 * t / (self.scr_efficiency_pct / 100.0 + 0.01)).min(2000.0); | support logic; im
| 269 | (blank) | blank separator; no runtime behavior. |
| 270 |         if self.dpf_soot_pct > 60.0 && self.aftertreatment == AftertreatmentState::Normal { | runtime gate; con
| 271 |             self.aftertreatment = AftertreatmentState::HighSoot; | support logic; impact depends on neighboring
| 272 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 273 |         if self.dpf_soot_pct > 80.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 274 |             self.aftertreatment = AftertreatmentState::Regenerating; | support logic; impact depends on neighbo
| 275 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 276 |         if self.dpf_soot_pct < 10.0 && self.aftertreatment == AftertreatmentState::Regenerating { | runtime gat
| 277 |             self.aftertreatment = AftertreatmentState::Normal; | support logic; impact depends on neighboring b
| 278 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 279 | (blank) | blank separator; no runtime behavior. |
| 280 |         // Electrical | comment/documentation; changing affects understanding and intent traceability. |
| 281 |         self.alternator_v = if self.rpm > 1000.0 { 14.2 } else { 12.4 }; | support logic; impact depends on nei
| 282 |         self.battery_v = if self.rpm > 1000.0 { 13.8 } else { 12.6 }; | support logic; impact depends on neighb
| 283 | (blank) | blank separator; no runtime behavior. |
| 284 |         // Service hours | comment/documentation; changing affects understanding and intent traceability. |
| 285 |         self.engine_hours += dt / 3600.0; | support logic; impact depends on neighboring block and data flow. |
| 286 |         self.service_hours_left -= dt / 3600.0; | support logic; impact depends on neighboring block and data f
| 287 | (blank) | blank separator; no runtime behavior. |
| 288 |         // Fault detection | comment/documentation; changing affects understanding and intent traceability. |
| 289 |         self.check_faults(); | support logic; impact depends on neighboring block and data flow. |
| 290 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 291 | (blank) | blank separator; no runtime behavior. |
| 292 |     fn check_faults(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 293 |         self.active_dtcs.retain(\|d\| d.active); | support logic; impact depends on neighboring block and data
| 294 |         self.mil_active = false; | support logic; impact depends on neighboring block and data flow. |
| 295 |         self.amber_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 296 |         self.red_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 297 | (blank) | blank separator; no runtime behavior. |
| 298 |         // High coolant temp → Red Stop | comment/documentation; changing affects understanding and intent trac
| 299 |         if self.coolant_temp_c > 108.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 300 |             self.inject_dtc( | support logic; impact depends on neighboring block and data flow. |
| 301 |                 110, | support logic; impact depends on neighboring block and data flow. |
| 302 |                 0, | support logic; impact depends on neighboring block and data flow. |
| 303 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 304 |                 "Engine Coolant Temp High — Red Stop", | support logic; impact depends on neighboring block and
| 305 |             ); | support logic; impact depends on neighboring block and data flow. |
| 306 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
```

```
| 307 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 308 |         // Low oil pressure → Red Stop | comment/documentation; changing affects understanding and intent trace
| 309 |         if self.oil_pressure_kpa < 80.0 { | runtime gate; condition changes can enable/disable critical paths.
| 310 |             self.inject_dtc( | support logic; impact depends on neighboring block and data flow. |
| 311 |                 100, | support logic; impact depends on neighboring block and data flow. |
| 312 |                 1, | support logic; impact depends on neighboring block and data flow. |
| 313 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 314 |                 "Engine Oil Pressure Low — Red Stop", | support logic; impact depends on neighboring block and
| 315 |             ); | support logic; impact depends on neighboring block and data flow. |
| 316 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 317 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 318 |         // Low DEF → Amber | comment/documentation; changing affects understanding and intent traceability. |
| 319 |         if self.def_level_pct < 10.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 320 |             self.inject_dtc( | support logic; impact depends on neighboring block and data flow. |
| 321 |                 3361, | support logic; impact depends on neighboring block and data flow. |
| 322 |                 17, | support logic; impact depends on neighboring block and data flow. |
| 323 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 324 |                 "DEF Level Low — SCR Derate Imminent", | support logic; impact depends on neighboring block and
| 325 |             ); | support logic; impact depends on neighboring block and data flow. |
| 326 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 327 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 328 |         // High soot → Amber | comment/documentation; changing affects understanding and intent traceability. |
| 329 |         if self.dpf_soot_pct > 70.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 330 |             self.inject_dtc( | support logic; impact depends on neighboring block and data flow. |
| 331 |                 3251, | support logic; impact depends on neighboring block and data flow. |
| 332 |                 15, | support logic; impact depends on neighboring block and data flow. |
| 333 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 334 |                 "DPF Soot Load High — Regen Required", | support logic; impact depends on neighboring block and
| 335 |             ); | support logic; impact depends on neighboring block and data flow. |
| 336 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 337 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 338 |         // Service overdue | comment/documentation; changing affects understanding and intent traceability. |
| 339 |         if self.service_hours_left < 0.0 { | runtime gate; condition changes can enable/disable critical paths.
| 340 |             self.inject_dtc(247, 31, DtcSeverity::Mil, "Service Interval Overdue"); | support logic; impact depe
| 341 |             self.mil_active = true; | support logic; impact depends on neighboring block and data flow. |
| 342 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 343 |         for dtc in &self.active_dtcs { | iteration logic; may alter timing, throughput, and safety constraints.
| 344 |             match dtc.severity { | branch dispatcher; arm changes alter behavior class handling. |
| 345 |                 DtcSeverity::Red => self.red_lamp = true, | support logic; impact depends on neighboring block
| 346 |                 DtcSeverity::Amber => self.amber_lamp = true, | support logic; impact depends on neighboring bl
| 347 |                 DtcSeverity::Mil => self.mil_active = true, | support logic; impact depends on neighboring bloc
| 348 |                 _ => {} | support logic; impact depends on neighboring block and data flow. |
| 349 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 350 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 351 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 352 | (blank) | blank separator; no runtime behavior. |
| 353 |     fn inject_dtc(&mut self, spn: u32, fmi: u8, sev: DtcSeverity, desc: &'static str) { | function signature/bo
| 354 |         if !self | runtime gate; condition changes can enable/disable critical paths. |
| 355 |             .active_dtcs | support logic; impact depends on neighboring block and data flow. |
| 356 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 357 |             .any(\|d\| d.spn == spn && d.fmi == fmi) | support logic; impact depends on neighboring block and d
| 358 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 359 |             self.active_dtcs.push(Dtc { | support logic; impact depends on neighboring block and data flow. |
| 360 |                 spn, | support logic; impact depends on neighboring block and data flow. |
| 361 |                 fmi, | support logic; impact depends on neighboring block and data flow. |
| 362 |                 count: 1, | support logic; impact depends on neighboring block and data flow. |
| 363 |                 active: true, | support logic; impact depends on neighboring block and data flow. |
| 364 |                 desc, | support logic; impact depends on neighboring block and data flow. |
| 365 |                 severity: sev, | support logic; impact depends on neighboring block and data flow. |
| 366 |             }); | support logic; impact depends on neighboring block and data flow. |
| 367 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 368 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 369 | (blank) | blank separator; no runtime behavior. |
| 370 |     pub fn clear_dtcs(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 371 |         for dtc in &mut self.active_dtcs { | iteration logic; may alter timing, throughput, and safety constrai
| 372 |             dtc.active = false; | support logic; impact depends on neighboring block and data flow. |
| 373 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 374 |         self.active_dtcs.retain(\|d\| d.active); | support logic; impact depends on neighboring block and data
| 375 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 376 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/ecm_mock_provider.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 198 | Non-empty: 189 | Symbol count: 4

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| build_mock_ecm_tree | 1 |
| len | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 6 | struct | MockParam | data/schema coupling and match/serde break risk |
| 13 | struct | MockFunction | data/schema coupling and match/serde break risk |
| 19 | struct | MockEcmNode | data/schema coupling and match/serde break risk |
| 25 | fn | build_mock_ecm_tree | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use auto_breaking::HeavyMachinery; | import wiring; changing path/name can break compilation and module linkage.
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | use crate::io::ecm_params::EcmSnapshot; | import wiring; changing path/name can break compilation and module linka
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | #[derive(Clone, Debug)] | comment/documentation; changing affects understanding and intent traceability. |
| 6 | pub struct MockParam { | data layout contract; field changes can break callers/serde/tests. |
| 7 |     pub key: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 8 |     pub value: f64, | support logic; impact depends on neighboring block and data flow. |
| 9 |     pub unit: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 10 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | #[derive(Clone, Debug)] | comment/documentation; changing affects understanding and intent traceability. |
| 13 | pub struct MockFunction { | data layout contract; field changes can break callers/serde/tests. |
| 14 |     pub name: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 15 |     pub params: Vec<MockParam>, | support logic; impact depends on neighboring block and data flow. |
| 16 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | #[derive(Clone, Debug)] | comment/documentation; changing affects understanding and intent traceability. |
| 19 | pub struct MockEcmNode { | data layout contract; field changes can break callers/serde/tests. |
| 20 |     pub name: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 21 |     pub source_address: u8, | support logic; impact depends on neighboring block and data flow. |
| 22 |     pub functions: Vec<MockFunction>, | support logic; impact depends on neighboring block and data flow. |
| 23 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 | pub fn build_mock_ecm_tree(bench: &HeavyMachinery, snap: &EcmSnapshot) -> Vec<MockEcmNode> { | function signature
| 26 |     let e = &bench.ecm; | support logic; impact depends on neighboring block and data flow. |
| 27 |     let h = &bench.hcm; | support logic; impact depends on neighboring block and data flow. |
| 28 |     let t = &bench.tcm; | support logic; impact depends on neighboring block and data flow. |
| 29 | (blank) | blank separator; no runtime behavior. |
| 30 |     let ecm = MockEcmNode { | support logic; impact depends on neighboring block and data flow. |
| 31 |         name: "Engine ECM", | support logic; impact depends on neighboring block and data flow. |
| 32 |         source_address: snap.source_address.unwrap_or(0x00), | support logic; impact depends on neighboring bloc
| 33 |         functions: vec![ | support logic; impact depends on neighboring block and data flow. |
| 34 |             MockFunction { | support logic; impact depends on neighboring block and data flow. |
| 35 |                 name: "Engine Performance", | support logic; impact depends on neighboring block and data flow.
| 36 |                 params: vec![ | support logic; impact depends on neighboring block and data flow. |
| 37 |                     MockParam { | support logic; impact depends on neighboring block and data flow. |
| 38 |                         key: "engine_speed_rpm", | support logic; impact depends on neighboring block and data f
| 39 |                         value: snap.engine_speed_rpm.unwrap_or(e.rpm), | support logic; impact depends on neighbo
| 40 |                         unit: "rpm", | support logic; impact depends on neighboring block and data flow. |
| 41 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 42 |                     MockParam { | support logic; impact depends on neighboring block and data flow. |
| 43 |                         key: "accel_pedal_pct", | support logic; impact depends on neighboring block and data flo
| 44 |                         value: snap.accel_pedal_pct.unwrap_or(e.active_throttle), | support logic; impact depends
| 45 |                         unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 46 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 47 |                     MockParam { | support logic; impact depends on neighboring block and data flow. |
| 48 |                         key: "actual_torque_nm", | support logic; impact depends on neighboring block and data f
| 49 |                         value: e.actual_torque_nm, | support logic; impact depends on neighboring block and data
| 50 |                         unit: "Nm", | support logic; impact depends on neighboring block and data flow. |
| 51 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 52 |                     MockParam { | support logic; impact depends on neighboring block and data flow. |
```

```
| 53 |                        key: "vehicle_speed", | support logic; impact depends on neighboring block and data flow
| 54 |                        value: t.ground_speed_kmh, | support logic; impact depends on neighboring block and data
| 55 |                        unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 56 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 57 |                ], | support logic; impact depends on neighboring block and data flow. |
| 58 |            }, | support logic; impact depends on neighboring block and data flow. |
| 59 |            MockFunction { | support logic; impact depends on neighboring block and data flow. |
| 60 |                name: "Pressures and Temperatures", | support logic; impact depends on neighboring block and data
| 61 |                params: vec![ | support logic; impact depends on neighboring block and data flow. |
| 62 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 63 |                        key: "coolant_temp_c", | support logic; impact depends on neighboring block and data flow
| 64 |                        value: snap.coolant_temp_c.unwrap_or(e.coolant_temp_c), | support logic; impact depends
| 65 |                        unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 66 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 67 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 68 |                        key: "fuel_temp_c", | support logic; impact depends on neighboring block and data flow.
| 69 |                        value: snap.fuel_temp_c.unwrap_or(e.fuel_temp_c), | support logic; impact depends on neig
| 70 |                        unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 71 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 72 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 73 |                        key: "oil_pressure_kpa", | support logic; impact depends on neighboring block and data f
| 74 |                        value: snap.oil_pressure_kpa.unwrap_or(e.oil_pressure_kpa), | support logic; impact depe
| 75 |                        unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 76 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 77 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 78 |                        key: "boost_pressure_kpa", | support logic; impact depends on neighboring block and data
| 79 |                        value: e.boost_pressure_kpa, | support logic; impact depends on neighboring block and da
| 80 |                        unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 81 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 82 |                ], | support logic; impact depends on neighboring block and data flow. |
| 83 |            }, | support logic; impact depends on neighboring block and data flow. |
| 84 |            MockFunction { | support logic; impact depends on neighboring block and data flow. |
| 85 |                name: "Aftertreatment", | support logic; impact depends on neighboring block and data flow. |
| 86 |                params: vec![ | support logic; impact depends on neighboring block and data flow. |
| 87 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 88 |                        key: "def_level_pct", | support logic; impact depends on neighboring block and data flow
| 89 |                        value: e.def_level_pct, | support logic; impact depends on neighboring block and data flo
| 90 |                        unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 91 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 92 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 93 |                        key: "dpf_soot_pct", | support logic; impact depends on neighboring block and data flow.
| 94 |                        value: e.dpf_soot_pct, | support logic; impact depends on neighboring block and data flow
| 95 |                        unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 96 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 97 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 98 |                        key: "scr_efficiency_pct", | support logic; impact depends on neighboring block and data
| 99 |                        value: e.scr_efficiency_pct, | support logic; impact depends on neighboring block and da
| 100 |                        unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 101 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 102 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 103 |                        key: "nox_tailpipe_ppm", | support logic; impact depends on neighboring block and data f
| 104 |                        value: e.nox_tailpipe_ppm, | support logic; impact depends on neighboring block and data
| 105 |                        unit: "ppm", | support logic; impact depends on neighboring block and data flow. |
| 106 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 107 |                ], | support logic; impact depends on neighboring block and data flow. |
| 108 |            }, | support logic; impact depends on neighboring block and data flow. |
| 109 |            MockFunction { | support logic; impact depends on neighboring block and data flow. |
| 110 |                name: "Diagnostics", | support logic; impact depends on neighboring block and data flow. |
| 111 |                params: vec![ | support logic; impact depends on neighboring block and data flow. |
| 112 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 113 |                        key: "active_dtcs", | support logic; impact depends on neighboring block and data flow.
| 114 |                        value: e.active_dtcs.len() as f64, | support logic; impact depends on neighboring block
| 115 |                        unit: "count", | support logic; impact depends on neighboring block and data flow. |
| 116 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 117 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 118 |                        key: "red_lamp", | support logic; impact depends on neighboring block and data flow. |
| 119 |                        value: if e.red_lamp { 1.0 } else { 0.0 }, | support logic; impact depends on neighbori
| 120 |                        unit: "bool", | support logic; impact depends on neighboring block and data flow. |
| 121 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 122 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 123 |                        key: "amber_lamp", | support logic; impact depends on neighboring block and data flow.
| 124 |                        value: if e.amber_lamp { 1.0 } else { 0.0 }, | support logic; impact depends on neighbo
| 125 |                        unit: "bool", | support logic; impact depends on neighboring block and data flow. |
| 126 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 127 |                    MockParam { | support logic; impact depends on neighboring block and data flow. |
| 128 |                        key: "mil_active", | support logic; impact depends on neighboring block and data flow.
```

```
| 129 |                         value: if e.mil_active { 1.0 } else { 0.0 }, | support logic; impact depends on neighbor
| 130 |                         unit: "bool", | support logic; impact depends on neighboring block and data flow. |
| 131 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 132 |                 ], | support logic; impact depends on neighboring block and data flow. |
| 133 |             }, | support logic; impact depends on neighboring block and data flow. |
| 134 |         ], | support logic; impact depends on neighboring block and data flow. |
| 135 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 136 | (blank) | blank separator; no runtime behavior. |
| 137 |     let hcm = MockEcmNode { | support logic; impact depends on neighboring block and data flow. |
| 138 |         name: "Hydraulic HCM", | support logic; impact depends on neighboring block and data flow. |
| 139 |         source_address: 0x1E, | support logic; impact depends on neighboring block and data flow. |
| 140 |         functions: vec![MockFunction { | support logic; impact depends on neighboring block and data flow. |
| 141 |             name: "Hydraulic", | support logic; impact depends on neighboring block and data flow. |
| 142 |             params: vec![ | support logic; impact depends on neighboring block and data flow. |
| 143 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 144 |                     key: "system_pressure_bar", | support logic; impact depends on neighboring block and data f
| 145 |                     value: h.system_pressure_bar, | support logic; impact depends on neighboring block and data
| 146 |                     unit: "bar", | support logic; impact depends on neighboring block and data flow. |
| 147 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 148 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 149 |                     key: "pump_flow_lpm", | support logic; impact depends on neighboring block and data flow. |
| 150 |                     value: h.pump_flow_lpm, | support logic; impact depends on neighboring block and data flow.
| 151 |                     unit: "L/min", | support logic; impact depends on neighboring block and data flow. |
| 152 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 153 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 154 |                     key: "fluid_temp_c", | support logic; impact depends on neighboring block and data flow. |
| 155 |                     value: h.fluid_temp_c, | support logic; impact depends on neighboring block and data flow.
| 156 |                     unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 157 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 158 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 159 |                     key: "filter_dp_bar", | support logic; impact depends on neighboring block and data flow. |
| 160 |                     value: h.filter_dp_bar, | support logic; impact depends on neighboring block and data flow.
| 161 |                     unit: "bar", | support logic; impact depends on neighboring block and data flow. |
| 162 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 163 |             ], | support logic; impact depends on neighboring block and data flow. |
| 164 |         }], | support logic; impact depends on neighboring block and data flow. |
| 165 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 | (blank) | blank separator; no runtime behavior. |
| 167 |     let tcm = MockEcmNode { | support logic; impact depends on neighboring block and data flow. |
| 168 |         name: "Transmission TCM", | support logic; impact depends on neighboring block and data flow. |
| 169 |         source_address: 0x03, | support logic; impact depends on neighboring block and data flow. |
| 170 |         functions: vec![MockFunction { | support logic; impact depends on neighboring block and data flow. |
| 171 |             name: "Driveline", | support logic; impact depends on neighboring block and data flow. |
| 172 |             params: vec![ | support logic; impact depends on neighboring block and data flow. |
| 173 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 174 |                     key: "ground_speed_kmh", | support logic; impact depends on neighboring block and data flow
| 175 |                     value: t.ground_speed_kmh, | support logic; impact depends on neighboring block and data flo
| 176 |                     unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 177 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 178 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 179 |                     key: "speed_step", | support logic; impact depends on neighboring block and data flow. |
| 180 |                     value: t.speed_step as f64, | support logic; impact depends on neighboring block and data f
| 181 |                     unit: "idx", | support logic; impact depends on neighboring block and data flow. |
| 182 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 183 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 184 |                     key: "clutch_slip_pct", | support logic; impact depends on neighboring block and data flow.
| 185 |                     value: t.clutch_slip_pct, | support logic; impact depends on neighboring block and data flo
| 186 |                     unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 187 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 188 |                 MockParam { | support logic; impact depends on neighboring block and data flow. |
| 189 |                     key: "gear_ratio", | support logic; impact depends on neighboring block and data flow. |
| 190 |                     value: t.gear_ratio, | support logic; impact depends on neighboring block and data flow. |
| 191 |                     unit: "ratio", | support logic; impact depends on neighboring block and data flow. |
| 192 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 193 |             ], | support logic; impact depends on neighboring block and data flow. |
| 194 |         }], | support logic; impact depends on neighboring block and data flow. |
| 195 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 196 | (blank) | blank separator; no runtime behavior. |
| 197 |     vec![ecm, tcm, hcm] | support logic; impact depends on neighboring block and data flow. |
| 198 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/ecu_abs.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 538 | Non-empty: 478 | Symbol count: 37

## Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| new | 9 |
| push | 3 |
| from_raw | 3 |
| build_id | 3 |
| fmt | 2 |
| default | 2 |
| update_v_ref | 2 |
| update_wheel_speeds | 2 |
| run_abs | 2 |
| run_tcs | 2 |
| run_esp | 2 |
| run_hill_hold | 2 |
| build_ebc1 | 2 |
| build_ebc2 | 2 |
| build_dm1_fault | 2 |
| is_locked | 1 |
| fraction | 1 |
| tick | 1 |
| toggle_abs | 1 |
| toggle_tcs | 1 |
| toggle_esp | 1 |
| any_wheel_abs_active | 1 |

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 19 | const | ABS_SA | namespace and compile-time linkage |
| 22 | const | FL | namespace and compile-time linkage |
| 23 | const | FR | namespace and compile-time linkage |
| 24 | const | RL | namespace and compile-time linkage |
| 25 | const | RR | namespace and compile-time linkage |
| 29 | enum | SensorState | data/schema coupling and match/serde break risk |
| 38 | enum | AbsValveState | data/schema coupling and match/serde break risk |
| 49 | impl | std::fmt::Display for AbsValveState | method behavior and trait semantics |
| 50 | fn | fmt | API and behavior coupling to callers/implementers |
| 62 | struct | Wheel | data/schema coupling and match/serde break risk |
| 85 | impl | Default for Wheel | method behavior and trait semantics |
| 86 | fn | default | API and behavior coupling to callers/implementers |
| 91 | impl | Wheel | method behavior and trait semantics |
| 92 | fn | new | API and behavior coupling to callers/implementers |
| 107 | fn | is_locked | API and behavior coupling to callers/implementers |
| 114 | enum | EspCondition | data/schema coupling and match/serde break risk |
| 124 | impl | std::fmt::Display for EspCondition | method behavior and trait semantics |
| 125 | fn | fmt | API and behavior coupling to callers/implementers |
| 137 | struct | EcuAbs | data/schema coupling and match/serde break risk |
| 190 | impl | Default for EcuAbs | method behavior and trait semantics |
| 191 | fn | default | API and behavior coupling to callers/implementers |
| 196 | impl | EcuAbs | method behavior and trait semantics |
| 197 | fn | new | API and behavior coupling to callers/implementers |
| 236 | fn | tick | API and behavior coupling to callers/implementers |
| 317 | fn | update_v_ref | API and behavior coupling to callers/implementers |
| 322 | fn | update_wheel_speeds | API and behavior coupling to callers/implementers |
| 342 | fn | run_abs | API and behavior coupling to callers/implementers |
| 393 | fn | run_tcs | API and behavior coupling to callers/implementers |
| 423 | fn | run_esp | API and behavior coupling to callers/implementers |
| 457 | fn | run_hill_hold | API and behavior coupling to callers/implementers |
| 467 | fn | build_ebc1 | API and behavior coupling to callers/implementers |
| 490 | fn | build_ebc2 | API and behavior coupling to callers/implementers |
| 508 | fn | build_dm1_fault | API and behavior coupling to callers/implementers |
| 525 | fn | toggle_abs | API and behavior coupling to callers/implementers |
| 528 | fn | toggle_tcs | API and behavior coupling to callers/implementers |
| 531 | fn | toggle_esp | API and behavior coupling to callers/implementers |
| 535 | fn | any_wheel_abs_active | API and behavior coupling to callers/implementers |

## Line by Line Change Impact

| Line | Code | What Happens If This Line Changes |

```
|---:|---|---|
| 1 | //! ABS / ESP / TCS — Electronic Stability Control ECU (SA 0x0B) | comment/documentation; changing affects unders
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Models a real stability control system: | comment/documentation; changing affects understanding and intent tra
| 4 | //!   ABS  — Anti-lock Braking System (ISO 11992, J1939 EBC1) | comment/documentation; changing affects understand
| 5 | //!   TCS  — Traction Control System (throttle + brake intervention) | comment/documentation; changing affects und
| 6 | //!   ESP  — Electronic Stability Program (yaw-rate based torque vectoring) | comment/documentation; changing affe
| 7 | //!   Hill Hold — Gradient hold via brake pressure | comment/documentation; changing affects understanding and int
| 8 | //!   EBS  — Electronic Brake System interface | comment/documentation; changing affects understanding and intent
| 9 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 10 | //! Each of the 4 wheels has: | comment/documentation; changing affects understanding and intent traceability. |
| 11 | //!   • Speed sensor (Hall-effect, active) | comment/documentation; changing affects understanding and intent tra
| 12 | //!   • Individual brake modulator (solenoid + pressure sensor) | comment/documentation; changing affects unders
| 13 | //!   • Slip calculation relative to reference speed | comment/documentation; changing affects understanding and
| 14 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 15 | //! J1939: Transmits EBC1 (20 ms), EBC2 (100 ms), DM1 (1 Hz) | comment/documentation; changing affects understand
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 | use crate::j1939::{self, addr, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 18 | (blank) | blank separator; no runtime behavior. |
| 19 | pub const ABS_SA: u8 = addr::BRAKES; // 0x0B | support logic; impact depends on neighboring block and data flow.
| 20 | (blank) | blank separator; no runtime behavior. |
| 21 | // ■■ Wheel index ■■■
```

```
| 76 |     pub tcs_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 77 |     /// Accumulated heat in brake disc (°C above ambient) | comment/documentation; changing affects understanding
| 78 |     pub brake_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 79 |     /// ABS cycle count for this wheel | comment/documentation; changing affects understanding and intent traceal
| 80 |     pub abs_cycles: u32, | support logic; impact depends on neighboring block and data flow. |
| 81 |     /// Internal ABS phase (0-1 sawtooth for pressure modulation) | comment/documentation; changing affects unde
| 82 |     abs_phase: f64, | support logic; impact depends on neighboring block and data flow. |
| 83 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 84 | (blank) | blank separator; no runtime behavior. |
| 85 | impl Default for Wheel { | implementation binding; behavior semantics and trait conformance live here. |
| 86 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 87 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 88 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 90 | (blank) | blank separator; no runtime behavior. |
| 91 | impl Wheel { | implementation binding; behavior semantics and trait conformance live here. |
| 92 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 93 |         Wheel { | support logic; impact depends on neighboring block and data flow. |
| 94 |             speed: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 95 |             slip_ratio: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 96 |             brake_pres_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 97 |             valve_state: AbsValveState::Normal, | support logic; impact depends on neighboring block and data flo
| 98 |             sensor_state: SensorState::Ok, | support logic; impact depends on neighboring block and data flow. |
| 99 |             abs_active: false, | support logic; impact depends on neighboring block and data flow. |
| 100 |             tcs_active: false, | support logic; impact depends on neighboring block and data flow. |
| 101 |             brake_temp_c: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 102 |             abs_cycles: 0, | support logic; impact depends on neighboring block and data flow. |
| 103 |             abs_phase: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 104 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 105 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 106 | (blank) | blank separator; no runtime behavior. |
| 107 |     pub fn is_locked(&self) -> bool { | function signature/body; changes affect API behavior and control-flow.
| 108 |         self.slip_ratio > 0.25 && self.speed < 1.0 | support logic; impact depends on neighboring block and data
| 109 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 110 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 111 | (blank) | blank separator; no runtime behavior. |
| 112 | // ■■ ESP Condition ■■■
```

```
| 152 |     pub v_ref_kmh: f64, | support logic; impact depends on neighboring block and data flow. |
| 153 | (blank) | blank separator; no runtime behavior. |
| 154 |     // ■ Brake system ■■■
```

```
| 228 |             t_dm1: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 229 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 230 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 231 | (blank) | blank separator; no runtime behavior. |
| 232 |     // ━━━
```

```
| 304 |             frames.push(self.build_ebc2(ts)); | support logic; impact depends on neighboring block and data flow
| 305 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 306 |         if self.t_dm1 >= 1.000 { | runtime gate; condition changes can enable/disable critical paths. |
| 307 |             self.t_dm1 = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 308 |             if self.abs_fault \|\| self.esp_fault { | runtime gate; condition changes can enable/disable critica
| 309 |                 frames.push(self.build_dm1_fault(ts)); | support logic; impact depends on neighboring block and
| 310 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 311 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 312 | (blank) | blank separator; no runtime behavior. |
| 313 |         (tcs_cut, frames) | support logic; impact depends on neighboring block and data flow. |
| 314 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 315 | (blank) | blank separator; no runtime behavior. |
| 316 |     // ━━━
```

```
| 380 |                     w.brake_pres_bar = pres; | support logic; impact depends on neighboring block and data flow
| 381 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 382 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 383 |                 w.valve_state = AbsValveState::Normal; | support logic; impact depends on neighboring block and
| 384 |                 w.brake_pres_bar = pres; | support logic; impact depends on neighboring block and data flow. |
| 385 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 386 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 387 | (blank) | blank separator; no runtime behavior. |
| 388 |         if self.abs_system_active { | runtime gate; condition changes can enable/disable critical paths. |
| 389 |             self.total_abs_events += 1; | support logic; impact depends on neighboring block and data flow. |
| 390 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 391 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 392 | (blank) | blank separator; no runtime behavior. |
| 393 |     fn run_tcs(&mut self, throttle_pct: f64, vehicle_speed: f64, _dt: f64) -> f64 { | function signature/body;
| 394 |         if !self.tcs_enabled { | runtime gate; condition changes can enable/disable critical paths. |
| 395 |             return 0.0; | support logic; impact depends on neighboring block and data flow. |
| 396 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 397 |         self.tcs_throttle_cut = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 398 | (blank) | blank separator; no runtime behavior. |
| 399 |         // TCS only active at low speed and high throttle | comment/documentation; changing affects understandi
| 400 |         if vehicle_speed > 40.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 401 |             return 0.0; | support logic; impact depends on neighboring block and data flow. |
| 402 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 403 | (blank) | blank separator; no runtime behavior. |
| 404 |         let driven_avg = (self.wheels[RL].speed + self.wheels[RR].speed) / 2.0; | support logic; impact depends
| 405 |         let spin_excess = driven_avg - vehicle_speed; | support logic; impact depends on neighboring block and
| 406 | (blank) | blank separator; no runtime behavior. |
| 407 |         if spin_excess > 3.0 && throttle_pct > 20.0 { | runtime gate; condition changes can enable/disable crit
| 408 |             let cut = ((spin_excess - 3.0) / 10.0).clamp(0.0, 0.8); | support logic; impact depends on neighbor
| 409 |             self.tcs_throttle_cut = cut; | support logic; impact depends on neighboring block and data flow. |
| 410 |             self.wheels[RL].tcs_active = true; | support logic; impact depends on neighboring block and data fl
| 411 |             self.wheels[RR].tcs_active = true; | support logic; impact depends on neighboring block and data fl
| 412 |             self.tcs_torque_request_nm = cut * 200.0; // request ECM to reduce torque | support logic; impact d
| 413 |             if self.tcs_system_active { | runtime gate; condition changes can enable/disable critical paths. |
| 414 |                 self.total_tcs_events += 1; | support logic; impact depends on neighboring block and data flow.
| 415 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 416 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 417 |             self.wheels[RL].tcs_active = false; | support logic; impact depends on neighboring block and data f
| 418 |             self.wheels[RR].tcs_active = false; | support logic; impact depends on neighboring block and data f
| 419 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 420 |         self.tcs_throttle_cut | support logic; impact depends on neighboring block and data flow. |
| 421 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 422 | (blank) | blank separator; no runtime behavior. |
| 423 |     fn run_esp(&mut self, speed: f64, steer_deg: f64, yaw_actual: f64, lat_g: f64, _dt: f64) { | function signat
| 424 |         if !self.esp_enabled \|\| speed < 15.0 { | runtime gate; condition changes can enable/disable critical
| 425 |             self.esp_condition = EspCondition::Neutral; | support logic; impact depends on neighboring block an
| 426 |             return; | support logic; impact depends on neighboring block and data flow. |
| 427 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 428 | (blank) | blank separator; no runtime behavior. |
| 429 |         // Desired yaw rate from bicycle model: v * steer / (L * (1 + Kus * v²)) | comment/documentation; chang
| 430 |         // L=2.7m, Kus=0.005 (understeer gradient) | comment/documentation; changing affects understanding and
| 431 |         let l = 2.7; | support logic; impact depends on neighboring block and data flow. |
| 432 |         let kus = 0.005; | support logic; impact depends on neighboring block and data flow. |
| 433 |         let v_ms = speed / 3.6; | support logic; impact depends on neighboring block and data flow. |
| 434 |         let steer_rad = steer_deg.to_radians(); | support logic; impact depends on neighboring block and data f
| 435 |         let yaw_desired = v_ms * steer_rad / (l * (1.0 + kus * v_ms * v_ms)); | support logic; impact depends o
| 436 |         let yaw_desired_degs = yaw_desired.to_degrees(); | support logic; impact depends on neighboring block a
| 437 | (blank) | blank separator; no runtime behavior. |
| 438 |         self.esp_yaw_error = yaw_actual - yaw_desired_degs; | support logic; impact depends on neighboring bloc
| 439 |         let err = self.esp_yaw_error.abs(); | support logic; impact depends on neighboring block and data flow.
| 440 | (blank) | blank separator; no runtime behavior. |
| 441 |         // Rollover detection: lateral acceleration > 0.8g | comment/documentation; changing affects understand
| 442 |         self.esp_condition = if lat_g.abs() > 0.8 { | support logic; impact depends on neighboring block and da
| 443 |             self.esp_system_active = true; | support logic; impact depends on neighboring block and data flow.
| 444 |             EspCondition::RolloverRisk | support logic; impact depends on neighboring block and data flow. |
| 445 |         } else if err > 8.0 && yaw_actual > yaw_desired_degs { | support logic; impact depends on neighboring b
| 446 |             self.esp_system_active = true; | support logic; impact depends on neighboring block and data flow.
| 447 |             EspCondition::Oversteer | support logic; impact depends on neighboring block and data flow. |
| 448 |         } else if err > 8.0 && yaw_actual < yaw_desired_degs { | support logic; impact depends on neighboring b
| 449 |             self.esp_system_active = true; | support logic; impact depends on neighboring block and data flow.
| 450 |             EspCondition::Understeer | support logic; impact depends on neighboring block and data flow. |
| 451 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 452 |             self.esp_system_active = false; | support logic; impact depends on neighboring block and data flow.
| 453 |             EspCondition::Neutral | support logic; impact depends on neighboring block and data flow. |
| 454 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 455 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 456 | (blank) | blank separator; no runtime behavior. |
| 457 |     fn run_hill_hold(&mut self, speed: f64, _dt: f64) { | function signature/body; changes affect API behavior a
| 458 |         // Activate hill hold if vehicle speed drops to 0 while brake applied | comment/documentation; changing
| 459 |         if speed < 0.5 && self.master_cylinder_bar > 10.0 { | runtime gate; condition changes can enable/disable
| 460 |             self.hill_hold_active = true; | support logic; impact depends on neighboring block and data flow. |
| 461 |         } else if speed > 2.0 { | support logic; impact depends on neighboring block and data flow. |
| 462 |             self.hill_hold_active = false; | support logic; impact depends on neighboring block and data flow.
| 463 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 464 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 465 | (blank) | blank separator; no runtime behavior. |
| 466 |     // ■ J1939 frame builders ■■■
```

```
| 532 |         self.esp_enabled = !self.esp_enabled; | support logic; impact depends on neighboring block and data flow
| 533 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 534 | (blank) | blank separator; no runtime behavior. |
| 535 |     pub fn any_wheel_abs_active(&self) -> bool { | function signature/body; changes affect API behavior and cont
| 536 |         self.wheels.iter().any(\|w\| w.abs_active) | support logic; impact depends on neighboring block and data
| 537 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 538 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/ecu_bcm.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 335 | Non-empty: 308 | Symbol count: 20

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 12 |
| check | 2 |
| default | 1 |
| tick | 1 |
| build_id | 1 |
| push | 1 |
| from_raw | 1 |
| toggle_work_lights | 1 |
| toggle_road_lights | 1 |
| toggle_beacon | 1 |
| honk | 1 |
| cycle_wiper | 1 |
| toggle_hvac | 1 |
| reset_all_fuses | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | const | BCM_SA | namespace and compile-time linkage |
| 11 | enum | WiperSpeed | data/schema coupling and match/serde break risk |
| 19 | enum | ChargingState | data/schema coupling and match/serde break risk |
| 26 | struct | Fuse | data/schema coupling and match/serde break risk |
| 33 | impl | Fuse | method behavior and trait semantics |
| 34 | fn | new | API and behavior coupling to callers/implementers |
| 42 | fn | check | API and behavior coupling to callers/implementers |
| 51 | struct | EcuBcm | data/schema coupling and match/serde break risk |
| 93 | impl | Default for EcuBcm | method behavior and trait semantics |
| 94 | fn | default | API and behavior coupling to callers/implementers |
| 99 | impl | EcuBcm | method behavior and trait semantics |
| 100 | fn | new | API and behavior coupling to callers/implementers |
| 139 | fn | tick | API and behavior coupling to callers/implementers |
| 304 | fn | toggle_work_lights | API and behavior coupling to callers/implementers |
| 308 | fn | toggle_road_lights | API and behavior coupling to callers/implementers |
| 311 | fn | toggle_beacon | API and behavior coupling to callers/implementers |
| 314 | fn | honk | API and behavior coupling to callers/implementers |
| 318 | fn | cycle_wiper | API and behavior coupling to callers/implementers |
| 326 | fn | toggle_hvac | API and behavior coupling to callers/implementers |
| 330 | fn | reset_all_fuses | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Body Control Module (BCM) — manages all electrical body functions. | comment/documentation; changing affects u
| 2 | //! SA: 0x27 (Cab Controller). | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Models: lighting, horn, wipers, battery/charging, fuses, switches. | comment/documentation; changing affects u
| 4 | //! J1939: Transmits heartbeat via Prop-B, monitors battery (PGN 65272). | comment/documentation; changing affects
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use crate::j1939::{self, addr, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | pub const BCM_SA: u8 = addr::CAB; // 0x27 | support logic; impact depends on neighboring block and data flow. |
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 11 | pub enum WiperSpeed { | state machine variants; changes ripple through matches and business logic. |
```

```
| 12 |     Off, | support logic; impact depends on neighboring block and data flow. |
| 13 |     Low, | support logic; impact depends on neighboring block and data flow. |
| 14 |     High, | support logic; impact depends on neighboring block and data flow. |
| 15 |     Intermittent, | support logic; impact depends on neighboring block and data flow. |
| 16 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 19 | pub enum ChargingState { | state machine variants; changes ripple through matches and business logic. |
| 20 |     Off, | support logic; impact depends on neighboring block and data flow. |
| 21 |     Charging, | support logic; impact depends on neighboring block and data flow. |
| 22 |     Fault, | support logic; impact depends on neighboring block and data flow. |
| 23 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 26 | pub struct Fuse { | data layout contract; field changes can break callers/serde/tests. |
| 27 |     pub id: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 28 |     pub rating_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 29 |     pub blown: bool, | support logic; impact depends on neighboring block and data flow. |
| 30 |     pub load_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 31 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 32 | (blank) | blank separator; no runtime behavior. |
| 33 | impl Fuse { | implementation binding; behavior semantics and trait conformance live here. |
| 34 |     pub fn new(id: &'static str, rating: f64) -> Self { | function signature/body; changes affect API behavior an
| 35 |         Fuse { | support logic; impact depends on neighboring block and data flow. |
| 36 |             id, | support logic; impact depends on neighboring block and data flow. |
| 37 |             rating_a: rating, | support logic; impact depends on neighboring block and data flow. |
| 38 |             blown: false, | support logic; impact depends on neighboring block and data flow. |
| 39 |             load_a: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 40 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 41 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 42 |     pub fn check(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 43 |         if self.load_a > self.rating_a * 1.15 { | runtime gate; condition changes can enable/disable critical pa
| 44 |             self.blown = true; | support logic; impact depends on neighboring block and data flow. |
| 45 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 46 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 | (blank) | blank separator; no runtime behavior. |
| 49 | // ■■ BCM ■■■
```

```
| 88 | (blank) | blank separator; no runtime behavior. |
| 89 |     // ■ J1939 TX timer ■■■
```

```
| 164 |             self.total_load_amps += 6.0; | support logic; impact depends on neighboring block and data flow. |
| 165 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 |         if self.cab_interior_light { | runtime gate; condition changes can enable/disable critical paths. |
| 167 |             self.total_load_amps += 2.0; | support logic; impact depends on neighboring block and data flow. |
| 168 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 169 |         if self.hvac_on { | runtime gate; condition changes can enable/disable critical paths. |
| 170 |             self.total_load_amps += 15.0; | support logic; impact depends on neighboring block and data flow. |
| 171 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 172 |         if self.horn_active { | runtime gate; condition changes can enable/disable critical paths. |
| 173 |             self.total_load_amps += 8.0; | support logic; impact depends on neighboring block and data flow. |
| 174 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 175 |         // Base ECU load | comment/documentation; changing affects understanding and intent traceability. |
| 176 |         self.total_load_amps += 5.0; | support logic; impact depends on neighboring block and data flow. |
| 177 | (blank) | blank separator; no runtime behavior. |
| 178 |         // ■ Battery current and voltage ■■■
```

```
| 240 |                 WiperSpeed::High => 1.4, | support logic; impact depends on neighboring block and data flow. |
| 241 |                 WiperSpeed::Intermittent => { | support logic; impact depends on neighboring block and data flow
| 242 |                     if (self.t_heartbeat % 3.0) < 0.5 { | runtime gate; condition changes can enable/disable cri
| 243 |                         1.0 | support logic; impact depends on neighboring block and data flow. |
| 244 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 245 |                         0.0 | support logic; impact depends on neighboring block and data flow. |
| 246 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 247 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 248 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 249 |             self.wiper_position += self.wiper_sweep_dir * sweep_speed * dt; | support logic; impact depends on n
| 250 |             if self.wiper_position >= 1.0 { | runtime gate; condition changes can enable/disable critical paths
| 251 |                 self.wiper_sweep_dir = -1.0; | support logic; impact depends on neighboring block and data flow
| 252 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 253 |             if self.wiper_position <= 0.0 { | runtime gate; condition changes can enable/disable critical paths
| 254 |                 self.wiper_sweep_dir = 1.0; | support logic; impact depends on neighboring block and data flow.
| 255 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 256 |             self.wiper_position = self.wiper_position.clamp(0.0, 1.0); | support logic; impact depends on neighb
| 257 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 258 | (blank) | blank separator; no runtime behavior. |
| 259 |         // ■ Horn (auto-off after 2 s) ■■■
```

```
| 316 |         self.horn_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 317 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 318 |     pub fn cycle_wiper(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 319 |         self.wiper_speed = match self.wiper_speed { | support logic; impact depends on neighboring block and dat
| 320 |             WiperSpeed::Off => WiperSpeed::Intermittent, | support logic; impact depends on neighboring block an
| 321 |             WiperSpeed::Intermittent => WiperSpeed::Low, | support logic; impact depends on neighboring block an
| 322 |             WiperSpeed::Low => WiperSpeed::High, | support logic; impact depends on neighboring block and data f
| 323 |             WiperSpeed::High => WiperSpeed::Off, | support logic; impact depends on neighboring block and data f
| 324 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 325 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 326 |     pub fn toggle_hvac(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 327 |         self.hvac_on = !self.hvac_on; | support logic; impact depends on neighboring block and data flow. |
| 328 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 329 | (blank) | blank separator; no runtime behavior. |
| 330 |     pub fn reset_all_fuses(&mut self) { | function signature/body; changes affect API behavior and control-flow
| 331 |         for fuse in &mut self.fuses { | iteration logic; may alter timing, throughput, and safety constraints.
| 332 |             fuse.blown = false; | support logic; impact depends on neighboring block and data flow. |
| 333 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 334 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 335 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/ecu_ecm.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 860 | Non-empty: 777 | Symbol count: 38

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| add_dtc | 17 |
| push | 10 |
| new | 5 |
| fmt | 3 |
| torque_at_rpm | 2 |
| update_aftertreatment | 2 |
| run_diagnostics | 2 |
| first_active_dtc_spn_fmi | 2 |
| default | 1 |
| tick | 1 |
| eec1 | 1 |
| eec2 | 1 |
| et1 | 1 |
| efl_p1 | 1 |
| ic1 | 1 |
| lfe | 1 |
| hours | 1 |
| dm1 | 1 |
| clear_dtcs | 1 |
| power_kw | 1 |
| acceleration_ms2 | 1 |
| is_running | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 17 | const | ECM_SA | namespace and compile-time linkage |
| 18 | const | CYLINDERS | namespace and compile-time linkage |
| 19 | const | IDLE_RPM | namespace and compile-time linkage |
| 20 | const | LO_IDLE_RPM | namespace and compile-time linkage |
| 21 | const | HI_IDLE_RPM | namespace and compile-time linkage |
| 22 | const | RATED_RPM | namespace and compile-time linkage |
| 23 | const | MAX_RPM | namespace and compile-time linkage |
| 24 | const | PEAK_TORQUE_NM | namespace and compile-time linkage |
| 25 | const | PEAK_TORQUE_RPM | namespace and compile-time linkage |
| 27 | const | RATED_POWER_KW | namespace and compile-time linkage |
| 28 | const | ENGINE_INERTIA_KGM2 | namespace and compile-time linkage |
| 29 | const | FUEL_TANK_L | namespace and compile-time linkage |
| 30 | const | DEF_TANK_L | namespace and compile-time linkage |
| 31 | const | DPF_REGEN_SOOT | namespace and compile-time linkage |
| 35 | enum | GovernorMode | data/schema coupling and match/serde break risk |
| 44 | impl | std::fmt::Display for GovernorMode | method behavior and trait semantics |
```

```
| 45 | fn | fmt | API and behavior coupling to callers/implementers |
| 56 | enum | FuelMap | data/schema coupling and match/serde break risk |
| 62 | impl | std::fmt::Display for FuelMap | method behavior and trait semantics |
| 63 | fn | fmt | API and behavior coupling to callers/implementers |
| 74 | enum | AftertreatmentState | data/schema coupling and match/serde break risk |
| 86 | impl | std::fmt::Display for AftertreatmentState | method behavior and trait semantics |
| 87 | fn | fmt | API and behavior coupling to callers/implementers |
| 100 | struct | EcuEcm | data/schema coupling and match/serde break risk |
| 196 | impl | Default for EcuEcm | method behavior and trait semantics |
| 197 | fn | default | API and behavior coupling to callers/implementers |
| 202 | impl | EcuEcm | method behavior and trait semantics |
| 203 | fn | new | API and behavior coupling to callers/implementers |
| 292 | fn | torque_at_rpm | API and behavior coupling to callers/implementers |
| 327 | fn | tick | API and behavior coupling to callers/implementers |
| 545 | fn | update_aftertreatment | API and behavior coupling to callers/implementers |
| 607 | fn | run_diagnostics | API and behavior coupling to callers/implementers |
| 806 | fn | add_dtc | API and behavior coupling to callers/implementers |
| 834 | fn | first_active_dtc_spn_fmi | API and behavior coupling to callers/implementers |
| 841 | fn | clear_dtcs | API and behavior coupling to callers/implementers |
| 850 | fn | power_kw | API and behavior coupling to callers/implementers |
| 853 | fn | acceleration_ms2 | API and behavior coupling to callers/implementers |
| 857 | fn | is_running | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Engine Control Module (ECM) — Tier 4 Final / Stage V diesel engine. | comment/documentation; changing affects
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Models a 6-cylinder, ~220 kW / 1050 Nm diesel with: | comment/documentation; changing affects understanding a
| 4 | //!   • Common Rail Direct Injection (CRDI) | comment/documentation; changing affects understanding and intent tra
| 5 | //!   • Variable Geometry Turbocharger (VGT) | comment/documentation; changing affects understanding and intent t
| 6 | //!   • Exhaust Gas Recirculation (EGR) | comment/documentation; changing affects understanding and intent traceal
| 7 | //!   • Diesel Particulate Filter (DPF) + SCR (DEF/AdBlue) | comment/documentation; changing affects understanding
| 8 | //!   • Electronic governor (all-speed, droop or low-idle/high-idle) | comment/documentation; changing affects un
| 9 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 10 | //! J1939 source address: 0x00 (Engine #1) | comment/documentation; changing affects understanding and intent tra
| 11 | //! Transmits: EEC1(10ms), EEC2(50ms), ET1(1000ms), EFL/P1(500ms), | comment/documentation; changing affects unde
| 12 | //!            IC1(500ms), LFE(100ms), HOURS(1000ms), DM1(1000ms) | comment/documentation; changing affects under
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 | use crate::j1939::{addr, Builder, Dtc, DtcSeverity, J1939Frame}; | import wiring; changing path/name can break c
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | // ■■ Engine constants ■■■
```

```
| 50 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 51 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 53 | (blank) | blank separator; no runtime behavior. |
| 54 | // ■■ Fuel Map Modes ■■■
```

```
| 126 |     pub active_throttle: f64,     // whichever is highest | support logic; impact depends on neighboring block a
| 127 | (blank) | blank separator; no runtime behavior. |
| 128 |     // ■ Fuel ■■■
```

```
| 202 | impl EcuEcm { | implementation binding; behavior semantics and trait conformance live here. |
| 203 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 204 |         EcuEcm { | support logic; impact depends on neighboring block and data flow. |
| 205 |             sa: ECM_SA, | support logic; impact depends on neighboring block and data flow. |
| 206 |             cylinders: CYLINDERS, | support logic; impact depends on neighboring block and data flow. |
| 207 |             fuel_map: FuelMap::Standard, | support logic; impact depends on neighboring block and data flow. |
| 208 |             governor_mode: GovernorMode::AllSpeed, | support logic; impact depends on neighboring block and data
| 209 |             engine_hours: 2347.5, | support logic; impact depends on neighboring block and data flow. |
| 210 |             service_interval_h: 500.0, | support logic; impact depends on neighboring block and data flow. |
| 211 |             service_hours_left: 152.5, | support logic; impact depends on neighboring block and data flow. |
| 212 | (blank) | blank separator; no runtime behavior. |
| 213 |             rpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 214 |             rpm_target: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 215 |             rpm_demand: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 216 |             rpm_limit: MAX_RPM, | support logic; impact depends on neighboring block and data flow. |
| 217 | (blank) | blank separator; no runtime behavior. |
| 218 |             actual_torque_nm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 219 |             percent_load: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 220 |             actual_torque_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 221 |             demand_torque_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 222 |             driveline_torque_nm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 223 | (blank) | blank separator; no runtime behavior. |
| 224 |             throttle_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 225 |             remote_throttle_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 226 |             active_throttle: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 227 | (blank) | blank separator; no runtime behavior. |
| 228 |             fuel_level_pct: 85.0, | support logic; impact depends on neighboring block and data flow. |
| 229 |             fuel_pressure_kpa: 450.0, | support logic; impact depends on neighboring block and data flow. |
| 230 |             fuel_rail_pressure_mpa: 180.0, | support logic; impact depends on neighboring block and data flow.
| 231 |             fuel_rate_lph: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 232 |             total_fuel_l: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 233 |             trip_fuel_l: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 234 | (blank) | blank separator; no runtime behavior. |
| 235 |             coolant_temp_c: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 236 |             oil_temp_c: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 237 |             fuel_temp_c: 22.0, | support logic; impact depends on neighboring block and data flow. |
| 238 |             intake_temp_c: 22.0, | support logic; impact depends on neighboring block and data flow. |
| 239 |             exhaust_temp_c: 150.0, | support logic; impact depends on neighboring block and data flow. |
| 240 |             turbo_oil_temp_c: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 241 |             ambient_temp_c: 22.0, | support logic; impact depends on neighboring block and data flow. |
| 242 | (blank) | blank separator; no runtime behavior. |
| 243 |             oil_pressure_kpa: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 244 |             oil_level_pct: 95.0, | support logic; impact depends on neighboring block and data flow. |
| 245 |             coolant_pres_kpa: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 246 |             boost_pressure_kpa: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 247 |             vgt_position_pct: 50.0, | support logic; impact depends on neighboring block and data flow. |
| 248 |             egr_valve_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 249 |             air_filter_dp_kpa: 1.2, | support logic; impact depends on neighboring block and data flow. |
| 250 |             intake_manifold_kpa: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 251 | (blank) | blank separator; no runtime behavior. |
| 252 |             battery_v: 12.8, | support logic; impact depends on neighboring block and data flow. |
| 253 |             alternator_v: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 254 |             starter_cranking: false, | support logic; impact depends on neighboring block and data flow. |
| 255 | (blank) | blank separator; no runtime behavior. |
| 256 |             def_level_pct: 72.0, | support logic; impact depends on neighboring block and data flow. |
| 257 |             def_quality_pct: 99.0, | support logic; impact depends on neighboring block and data flow. |
| 258 |             dpf_soot_pct: 18.0, | support logic; impact depends on neighboring block and data flow. |
| 259 |             dpf_temp_c: 200.0, | support logic; impact depends on neighboring block and data flow. |
| 260 |             scr_inlet_temp_c: 200.0, | support logic; impact depends on neighboring block and data flow. |
| 261 |             scr_efficiency_pct: 95.0, | support logic; impact depends on neighboring block and data flow. |
| 262 |             nox_raw_ppm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 263 |             nox_tailpipe_ppm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 264 |             aftertreatment: AftertreatmentState::Normal, | support logic; impact depends on neighboring block a
| 265 |             regen_requested: false, | support logic; impact depends on neighboring block and data flow. |
| 266 |             regen_inhibited: false, | support logic; impact depends on neighboring block and data flow. |
| 267 | (blank) | blank separator; no runtime behavior. |
| 268 |             active_dtcs: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 269 |             stored_dtcs: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 270 |             mil_active: false, | support logic; impact depends on neighboring block and data flow. |
| 271 |             amber_lamp: false, | support logic; impact depends on neighboring block and data flow. |
| 272 |             red_lamp: false, | support logic; impact depends on neighboring block and data flow. |
| 273 |             protect_lamp: false, | support logic; impact depends on neighboring block and data flow. |
| 274 | (blank) | blank separator; no runtime behavior. |
| 275 |             t_eec1: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 276 |             t_eec2: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 277 |             t_et1: 0.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 278 |             t_efl: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 279 |             t_ic1: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 280 |             t_lfe: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 281 |             t_hrs: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 282 |             t_dm1: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 283 | (blank) | blank separator; no runtime behavior. |
| 284 |             rpm_error_integral: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 285 |             warmup_done: false, | support logic; impact depends on neighboring block and data flow. |
| 286 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 287 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 288 | (blank) | blank separator; no runtime behavior. |
| 289 |     // ━━━
```

```
| 354 |         self.rpm_error_integral = (self.rpm_error_integral + rpm_err * dt).clamp(-500.0, 500.0); | support logic
| 355 |         // Governor correction (PI) | comment/documentation; changing affects understanding and intent traceabil
| 356 |         let governor_correction = rpm_err * 0.8 + self.rpm_error_integral * 0.3; | support logic; impact depends
| 357 |         self.rpm += (rpm_accel * 0.05 + governor_correction) * dt; | support logic; impact depends on neighborin
| 358 |         self.rpm = self.rpm.clamp(0.0, MAX_RPM); | support logic; impact depends on neighboring block and data f
| 359 | (blank) | blank separator; no runtime behavior. |
| 360 |         // Engine stalls if RPM drops below threshold with no throttle | comment/documentation; changing affects
| 361 |         if self.rpm < 400.0 && self.active_throttle < 2.0 { | runtime gate; condition changes can enable/disable
| 362 |             self.rpm = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 363 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 364 | (blank) | blank separator; no runtime behavior. |
| 365 |         // ■ Torque and load computation ■■■
```

```
| 430 |         } else if self.rpm > 400.0 { | support logic; impact depends on neighboring block and data flow. |
| 431 |             13.0 | support logic; impact depends on neighboring block and data flow. |
| 432 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 433 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 434 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 435 |         self.battery_v = if self.alternator_v > 13.0 { | support logic; impact depends on neighboring block and
| 436 |             13.8 | support logic; impact depends on neighboring block and data flow. |
| 437 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 438 |             (self.battery_v - 0.001 * dt).max(11.5) | support logic; impact depends on neighboring block and da
| 439 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 440 | (blank) | blank separator; no runtime behavior. |
| 441 |         // ■ Aftertreatment ■■■
```

```
| 506 |             self.t_ic1 = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 507 |             frames.push(Builder::ic1( | support logic; impact depends on neighboring block and data flow. |
| 508 |                 ts, | support logic; impact depends on neighboring block and data flow. |
| 509 |                 self.boost_pressure_kpa, | support logic; impact depends on neighboring block and data flow. |
| 510 |                 self.intake_temp_c, | support logic; impact depends on neighboring block and data flow. |
| 511 |                 self.exhaust_temp_c, | support logic; impact depends on neighboring block and data flow. |
| 512 |                 self.sa, | support logic; impact depends on neighboring block and data flow. |
| 513 |             )); | support logic; impact depends on neighboring block and data flow. |
| 514 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 515 |         if self.t_lfe >= 0.100 { | runtime gate; condition changes can enable/disable critical paths. |
| 516 |             self.t_lfe = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 517 |             frames.push(Builder::lfe( | support logic; impact depends on neighboring block and data flow. |
| 518 |                 ts, | support logic; impact depends on neighboring block and data flow. |
| 519 |                 self.fuel_rate_lph, | support logic; impact depends on neighboring block and data flow. |
| 520 |                 self.active_throttle, | support logic; impact depends on neighboring block and data flow. |
| 521 |                 self.sa, | support logic; impact depends on neighboring block and data flow. |
| 522 |             )); | support logic; impact depends on neighboring block and data flow. |
| 523 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 524 |         if self.t_hrs >= 1.000 { | runtime gate; condition changes can enable/disable critical paths. |
| 525 |             self.t_hrs = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 526 |             frames.push(Builder::hours(ts, self.engine_hours, self.sa)); | support logic; impact depends on neig
| 527 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 528 |         if self.t_dm1 >= 1.000 { | runtime gate; condition changes can enable/disable critical paths. |
| 529 |             self.t_dm1 = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 530 |             let (spn, fmi) = self.first_active_dtc_spn_fmi(); | support logic; impact depends on neighboring blo
| 531 |             frames.push(Builder::dm1( | support logic; impact depends on neighboring block and data flow. |
| 532 |                 ts, | support logic; impact depends on neighboring block and data flow. |
| 533 |                 self.amber_lamp, | support logic; impact depends on neighboring block and data flow. |
| 534 |                 self.red_lamp, | support logic; impact depends on neighboring block and data flow. |
| 535 |                 self.mil_active, | support logic; impact depends on neighboring block and data flow. |
| 536 |                 spn, | support logic; impact depends on neighboring block and data flow. |
| 537 |                 fmi, | support logic; impact depends on neighboring block and data flow. |
| 538 |                 self.sa, | support logic; impact depends on neighboring block and data flow. |
| 539 |             )); | support logic; impact depends on neighboring block and data flow. |
| 540 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 541 | (blank) | blank separator; no runtime behavior. |
| 542 |         frames | support logic; impact depends on neighboring block and data flow. |
| 543 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 544 | (blank) | blank separator; no runtime behavior. |
| 545 |     fn update_aftertreatment(&mut self, dt: f64) { | function signature/body; changes affect API behavior and co
| 546 |         let load = self.percent_load / 100.0; | support logic; impact depends on neighboring block and data flow
| 547 |         let rpm_running = self.rpm > 400.0; | support logic; impact depends on neighboring block and data flow.
| 548 | (blank) | blank separator; no runtime behavior. |
| 549 |         // DPF soot accumulates with load, decreases during regen | comment/documentation; changing affects unde
| 550 |         if rpm_running { | runtime gate; condition changes can enable/disable critical paths. |
| 551 |             let soot_rate = match self.aftertreatment { | support logic; impact depends on neighboring block and
| 552 |                 AftertreatmentState::DpfRegen => -3.0 * dt, // burn soot at 3%/s | support logic; impact depends
| 553 |                 _ => 0.0008 * load * dt, | support logic; impact depends on neighboring block and data flow. |
| 554 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 555 |             self.dpf_soot_pct = (self.dpf_soot_pct + soot_rate).clamp(0.0, 100.0); | support logic; impact depe
| 556 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 557 | (blank) | blank separator; no runtime behavior. |
| 558 |         // DPF temperature | comment/documentation; changing affects understanding and intent traceability. |
| 559 |         let dpf_target = if self.aftertreatment == AftertreatmentState::DpfRegen { | support logic; impact depe
| 560 |             620.0 | support logic; impact depends on neighboring block and data flow. |
| 561 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 562 |             200.0 + self.exhaust_temp_c * 0.5 | support logic; impact depends on neighboring block and data flo
| 563 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 564 |         self.dpf_temp_c += (dpf_target - self.dpf_temp_c) * dt * 0.05; | support logic; impact depends on neigh
| 565 | (blank) | blank separator; no runtime behavior. |
| 566 |         // SCR inlet temperature | comment/documentation; changing affects understanding and intent traceabilit
| 567 |         self.scr_inlet_temp_c = self.dpf_temp_c * 0.85; | support logic; impact depends on neighboring block an
| 568 | (blank) | blank separator; no runtime behavior. |
| 569 |         // SCR efficiency depends on temperature and DEF availability | comment/documentation; changing affects
| 570 |         let scr_temp_ok = self.scr_inlet_temp_c > 200.0; | support logic; impact depends on neighboring block a
| 571 |         self.scr_efficiency_pct = if self.def_level_pct > 5.0 && scr_temp_ok { | support logic; impact depends
| 572 |             93.0 + load * 4.0 | support logic; impact depends on neighboring block and data flow. |
| 573 |         } else if self.def_level_pct > 2.0 { | support logic; impact depends on neighboring block and data flow
| 574 |             40.0 | support logic; impact depends on neighboring block and data flow. |
| 575 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 576 |             5.0 // near zero DEF | support logic; impact depends on neighboring block and data flow. |
| 577 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 578 | (blank) | blank separator; no runtime behavior. |
| 579 |         // DEF consumption: ~3-5% of fuel consumed | comment/documentation; changing affects understanding and i
| 580 |         let def_consumed = self.fuel_rate_lph * 0.04 / 3600.0 * dt; | support logic; impact depends on neighbori
| 581 |         self.def_level_pct = (self.def_level_pct - def_consumed / DEF_TANK_L * 100.0).max(0.0); | support logic
```

```
| 582 | (blank) | blank separator; no runtime behavior. |
| 583 |         // NOx | comment/documentation; changing affects understanding and intent traceability. |
| 584 |         self.nox_raw_ppm = if rpm_running { | support logic; impact depends on neighboring block and data flow.
| 585 |             800.0 + load * 1200.0 * (1.0 - self.egr_valve_pct / 80.0) | support logic; impact depends on neighbo
| 586 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 587 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 588 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 589 |         self.nox_tailpipe_ppm = self.nox_raw_ppm * (1.0 - self.scr_efficiency_pct / 100.0); | support logic; imp
| 590 | (blank) | blank separator; no runtime behavior. |
| 591 |         // Aftertreatment state transitions | comment/documentation; changing affects understanding and intent t
| 592 |         self.aftertreatment = if self.dpf_soot_pct > DPF_REGEN_SOOT && !self.regen_inhibited { | support logic;
| 593 |             self.regen_requested = true; | support logic; impact depends on neighboring block and data flow. |
| 594 |             AftertreatmentState::DpfRegen | support logic; impact depends on neighboring block and data flow. |
| 595 |         } else if self.dpf_soot_pct < 5.0 { | support logic; impact depends on neighboring block and data flow.
| 596 |             self.regen_requested = false; | support logic; impact depends on neighboring block and data flow. |
| 597 |             AftertreatmentState::Normal | support logic; impact depends on neighboring block and data flow. |
| 598 |         } else if self.def_level_pct < 2.0 { | support logic; impact depends on neighboring block and data flow
| 599 |             AftertreatmentState::Derating | support logic; impact depends on neighboring block and data flow. |
| 600 |         } else if self.scr_efficiency_pct < 50.0 { | support logic; impact depends on neighboring block and data
| 601 |             AftertreatmentState::ScrDegraded | support logic; impact depends on neighboring block and data flow
| 602 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 603 |             self.aftertreatment | support logic; impact depends on neighboring block and data flow. |
| 604 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 605 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 606 | (blank) | blank separator; no runtime behavior. |
| 607 |     fn run_diagnostics(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 608 |         self.active_dtcs.retain(\|d\| d.active); | support logic; impact depends on neighboring block and data
| 609 |         self.mil_active = false; | support logic; impact depends on neighboring block and data flow. |
| 610 |         self.amber_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 611 |         self.red_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 612 |         self.protect_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 613 | (blank) | blank separator; no runtime behavior. |
| 614 |         // SPN 110: Engine Coolant Temperature | comment/documentation; changing affects understanding and inter
| 615 |         if self.coolant_temp_c > 107.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 616 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 617 |                 110, | support logic; impact depends on neighboring block and data flow. |
| 618 |                 0, | support logic; impact depends on neighboring block and data flow. |
| 619 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 620 |                 "Coolant temp above normal — Red Stop", | support logic; impact depends on neighboring block and
| 621 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 622 |             ); | support logic; impact depends on neighboring block and data flow. |
| 623 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 624 |         } else if self.coolant_temp_c > 102.0 { | support logic; impact depends on neighboring block and data f
| 625 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 626 |                 110, | support logic; impact depends on neighboring block and data flow. |
| 627 |                 15, | support logic; impact depends on neighboring block and data flow. |
| 628 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 629 |                 "Coolant temp high — Amber Warning", | support logic; impact depends on neighboring block and d
| 630 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 631 |             ); | support logic; impact depends on neighboring block and data flow. |
| 632 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 633 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 634 | (blank) | blank separator; no runtime behavior. |
| 635 |         // SPN 100: Engine Oil Pressure | comment/documentation; changing affects understanding and intent trac
| 636 |         if self.rpm > 600.0 && self.oil_pressure_kpa < 80.0 { | runtime gate; condition changes can enable/disa
| 637 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 638 |                 100, | support logic; impact depends on neighboring block and data flow. |
| 639 |                 1, | support logic; impact depends on neighboring block and data flow. |
| 640 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 641 |                 "Engine oil pressure critically low — Red Stop", | support logic; impact depends on neighboring
| 642 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 643 |             ); | support logic; impact depends on neighboring block and data flow. |
| 644 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 645 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 646 | (blank) | blank separator; no runtime behavior. |
| 647 |         // SPN 183: Fuel rate (SPN 94 = fuel delivery pressure) | comment/documentation; changing affects under
| 648 |         if self.fuel_pressure_kpa < 150.0 && self.rpm > 600.0 { | runtime gate; condition changes can enable/di
| 649 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 650 |                 94, | support logic; impact depends on neighboring block and data flow. |
| 651 |                 1, | support logic; impact depends on neighboring block and data flow. |
| 652 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 653 |                 "Fuel delivery pressure low — check fuel filter", | support logic; impact depends on neighborin
| 654 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 655 |             ); | support logic; impact depends on neighboring block and data flow. |
| 656 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 657 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 658 | (blank) | blank separator; no runtime behavior. |
| 659 |         // SPN 3361: DEF Level | comment/documentation; changing affects understanding and intent traceability.
| 660 |         if self.def_level_pct < 10.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 661 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 662 |                 3361, | support logic; impact depends on neighboring block and data flow. |
| 663 |                 17, | support logic; impact depends on neighboring block and data flow. |
| 664 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 665 |                 "DEF level low — refill before derating", | support logic; impact depends on neighboring block a
| 666 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 667 |             ); | support logic; impact depends on neighboring block and data flow. |
| 668 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 669 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 670 |         if self.def_level_pct < 2.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 671 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 672 |                 3361, | support logic; impact depends on neighboring block and data flow. |
| 673 |                 1, | support logic; impact depends on neighboring block and data flow. |
| 674 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 675 |                 "DEF level critically low — power derate active", | support logic; impact depends on neighboring
| 676 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 677 |             ); | support logic; impact depends on neighboring block and data flow. |
| 678 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 679 |             self.rpm_limit = RATED_RPM * 0.60; // 60% derating | support logic; impact depends on neighboring bl
| 680 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 681 |             self.rpm_limit = MAX_RPM; | support logic; impact depends on neighboring block and data flow. |
| 682 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 683 | (blank) | blank separator; no runtime behavior. |
| 684 |         // SPN 3251: DPF Soot Load | comment/documentation; changing affects understanding and intent traceabil
| 685 |         if self.dpf_soot_pct > 80.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 686 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 687 |                 3251, | support logic; impact depends on neighboring block and data flow. |
| 688 |                 16, | support logic; impact depends on neighboring block and data flow. |
| 689 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 690 |                 "DPF soot load high — manual regen required", | support logic; impact depends on neighboring bl
| 691 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 692 |             ); | support logic; impact depends on neighboring block and data flow. |
| 693 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 694 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 695 | (blank) | blank separator; no runtime behavior. |
| 696 |         // SPN 102: Boost / Intake manifold pressure | comment/documentation; changing affects understanding an
| 697 |         if self.boost_pressure_kpa > 320.0 \|\| self.intake_manifold_kpa > 320.0 { | runtime gate; condition cha
| 698 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 699 |                 102, | support logic; impact depends on neighboring block and data flow. |
| 700 |                 0, | support logic; impact depends on neighboring block and data flow. |
| 701 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 702 |                 "Boost pressure above safe limit — inspect VGT/wastegate", | support logic; impact depends on n
| 703 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 704 |             ); | support logic; impact depends on neighboring block and data flow. |
| 705 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 706 |         } else if self.rpm > 1000.0 && self.boost_pressure_kpa < 95.0 { | support logic; impact depends on neigh
| 707 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 708 |                 102, | support logic; impact depends on neighboring block and data flow. |
| 709 |                 1, | support logic; impact depends on neighboring block and data flow. |
| 710 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 711 |                 "Boost pressure below expected level — possible leak or turbo issue", | support logic; impact de
| 712 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 713 |             ); | support logic; impact depends on neighboring block and data flow. |
| 714 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 715 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 716 | (blank) | blank separator; no runtime behavior. |
| 717 |         // SPN 111: Coolant pressure low | comment/documentation; changing affects understanding and intent trac
| 718 |         if self.rpm > 600.0 && self.coolant_pres_kpa < 55.0 { | runtime gate; condition changes can enable/disal
| 719 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 720 |                 111, | support logic; impact depends on neighboring block and data flow. |
| 721 |                 1, | support logic; impact depends on neighboring block and data flow. |
| 722 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 723 |                 "Coolant pressure low — inspect cap/pump/leak", | support logic; impact depends on neighboring b
| 724 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 725 |             ); | support logic; impact depends on neighboring block and data flow. |
| 726 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 727 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 728 | (blank) | blank separator; no runtime behavior. |
| 729 |         // SPN 105: Intake manifold temperature high | comment/documentation; changing affects understanding an
| 730 |         if self.intake_temp_c > 95.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 731 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 732 |                 105, | support logic; impact depends on neighboring block and data flow. |
| 733 |                 15, | support logic; impact depends on neighboring block and data flow. |
```

```
| 734 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 735 |                 "Intake manifold temperature high — charge-air cooling degraded", | support logic; impact depend
| 736 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 737 |             ); | support logic; impact depends on neighboring block and data flow. |
| 738 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 739 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 740 | (blank) | blank separator; no runtime behavior. |
| 741 |         // SPN 173: Exhaust temperature severe | comment/documentation; changing affects understanding and inter
| 742 |         if self.exhaust_temp_c > 820.0 \|\| self.dpf_temp_c > 760.0 { | runtime gate; condition changes can enal
| 743 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 744 |                 173, | support logic; impact depends on neighboring block and data flow. |
| 745 |                 0, | support logic; impact depends on neighboring block and data flow. |
| 746 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 747 |                 "Exhaust temperature above severe threshold", | support logic; impact depends on neighboring bl
| 748 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 749 |             ); | support logic; impact depends on neighboring block and data flow. |
| 750 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 751 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 752 | (blank) | blank separator; no runtime behavior. |
| 753 |         // SPN 1761: DEF quality abnormal | comment/documentation; changing affects understanding and intent tra
| 754 |         if self.def_quality_pct < 80.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 755 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 756 |                 1761, | support logic; impact depends on neighboring block and data flow. |
| 757 |                 3, | support logic; impact depends on neighboring block and data flow. |
| 758 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 759 |                 "DEF quality out of expected range", | support logic; impact depends on neighboring block and da
| 760 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 761 |             ); | support logic; impact depends on neighboring block and data flow. |
| 762 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 763 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 764 | (blank) | blank separator; no runtime behavior. |
| 765 |         // SPN 3226: Tailpipe NOx excessive | comment/documentation; changing affects understanding and intent t
| 766 |         if self.nox_tailpipe_ppm > 900.0 { | runtime gate; condition changes can enable/disable critical paths.
| 767 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 768 |                 3226, | support logic; impact depends on neighboring block and data flow. |
| 769 |                 16, | support logic; impact depends on neighboring block and data flow. |
| 770 |                 DtcSeverity::Red, | support logic; impact depends on neighboring block and data flow. |
| 771 |                 "Tailpipe NOx above legal threshold", | support logic; impact depends on neighboring block and d
| 772 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 773 |             ); | support logic; impact depends on neighboring block and data flow. |
| 774 |             self.red_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 775 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 776 | (blank) | blank separator; no runtime behavior. |
| 777 |         // SPN 157: Common rail pressure low | comment/documentation; changing affects understanding and intent
| 778 |         if self.rpm > 900.0 && self.fuel_rail_pressure_mpa < 70.0 { | runtime gate; condition changes can enabl
| 779 |             self.add_dtc( | support logic; impact depends on neighboring block and data flow. |
| 780 |                 157, | support logic; impact depends on neighboring block and data flow. |
| 781 |                 1, | support logic; impact depends on neighboring block and data flow. |
| 782 |                 DtcSeverity::Amber, | support logic; impact depends on neighboring block and data flow. |
| 783 |                 "Fuel rail pressure below expected range", | support logic; impact depends on neighboring block
| 784 |                 true, | support logic; impact depends on neighboring block and data flow. |
| 785 |             ); | support logic; impact depends on neighboring block and data flow. |
| 786 |             self.amber_lamp = true; | support logic; impact depends on neighboring block and data flow. |
| 787 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 788 | (blank) | blank separator; no runtime behavior. |
| 789 |         // SPN 247: Service interval | comment/documentation; changing affects understanding and intent traceab
| 790 |         if self.service_hours_left < 0.0 { | runtime gate; condition changes can enable/disable critical paths.
| 791 |             self.add_dtc(247, 31, DtcSeverity::Mil, "Service interval overdue", true); | support logic; impact
| 792 |             self.mil_active = true; | support logic; impact depends on neighboring block and data flow. |
| 793 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 794 | (blank) | blank separator; no runtime behavior. |
| 795 |         // Propagate lamp states from DTC list | comment/documentation; changing affects understanding and inter
| 796 |         for dtc in &self.active_dtcs { | iteration logic; may alter timing, throughput, and safety constraints.
| 797 |             match dtc.severity { | branch dispatcher; arm changes alter behavior class handling. |
| 798 |                 DtcSeverity::Red => self.red_lamp = true, | support logic; impact depends on neighboring block a
| 799 |                 DtcSeverity::Amber => self.amber_lamp = true, | support logic; impact depends on neighboring bl
| 800 |                 DtcSeverity::Mil => self.mil_active = true, | support logic; impact depends on neighboring bloc
| 801 |                 DtcSeverity::Protect => self.protect_lamp = true, | support logic; impact depends on neighboring
| 802 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 803 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 804 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 805 | (blank) | blank separator; no runtime behavior. |
| 806 |     fn add_dtc(&mut self, spn: u32, fmi: u8, sev: DtcSeverity, desc: &'static str, active: bool) { | function s
| 807 |         if !self | runtime gate; condition changes can enable/disable critical paths. |
| 808 |             .active_dtcs | support logic; impact depends on neighboring block and data flow. |
| 809 |             .iter() | support logic; impact depends on neighboring block and data flow. |
```

```
| 810 |             .any(\|d\| d.spn == spn && d.fmi == fmi) | support logic; impact depends on neighboring block and da
| 811 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 812 |             // Move to stored if was previously active | comment/documentation; changing affects understanding a
| 813 |             self.stored_dtcs.push(Dtc { | support logic; impact depends on neighboring block and data flow. |
| 814 |                 spn, | support logic; impact depends on neighboring block and data flow. |
| 815 |                 fmi, | support logic; impact depends on neighboring block and data flow. |
| 816 |                 count: 1, | support logic; impact depends on neighboring block and data flow. |
| 817 |                 active: false, | support logic; impact depends on neighboring block and data flow. |
| 818 |                 desc, | support logic; impact depends on neighboring block and data flow. |
| 819 |                 severity: sev, | support logic; impact depends on neighboring block and data flow. |
| 820 |             }); | support logic; impact depends on neighboring block and data flow. |
| 821 |             if active { | runtime gate; condition changes can enable/disable critical paths. |
| 822 |                 self.active_dtcs.push(Dtc { | support logic; impact depends on neighboring block and data flow.
| 823 |                     spn, | support logic; impact depends on neighboring block and data flow. |
| 824 |                     fmi, | support logic; impact depends on neighboring block and data flow. |
| 825 |                     count: 1, | support logic; impact depends on neighboring block and data flow. |
| 826 |                     active: true, | support logic; impact depends on neighboring block and data flow. |
| 827 |                     desc, | support logic; impact depends on neighboring block and data flow. |
| 828 |                     severity: sev, | support logic; impact depends on neighboring block and data flow. |
| 829 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 830 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 831 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 832 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 833 | (blank) | blank separator; no runtime behavior. |
| 834 |     fn first_active_dtc_spn_fmi(&self) -> (u32, u8) { | function signature/body; changes affect API behavior and
| 835 |         self.active_dtcs | support logic; impact depends on neighboring block and data flow. |
| 836 |             .first() | support logic; impact depends on neighboring block and data flow. |
| 837 |             .map(\|d\| (d.spn, d.fmi)) | support logic; impact depends on neighboring block and data flow. |
| 838 |             .unwrap_or((0, 0)) | support logic; impact depends on neighboring block and data flow. |
| 839 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 840 | (blank) | blank separator; no runtime behavior. |
| 841 |     pub fn clear_dtcs(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 842 |         self.active_dtcs.clear(); | support logic; impact depends on neighboring block and data flow. |
| 843 |         self.mil_active = false; | support logic; impact depends on neighboring block and data flow. |
| 844 |         self.amber_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 845 |         self.red_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 846 |         self.protect_lamp = false; | support logic; impact depends on neighboring block and data flow. |
| 847 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 848 | (blank) | blank separator; no runtime behavior. |
| 849 |     // ■ Status helpers ■■■
```

```
| set_loader_lift_cmd | 1 |
| set_loader_tilt_cmd | 1 |
| set_aux_cmd | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 16 | const | HCM_SA | namespace and compile-time linkage |
| 20 | enum | PumpMode | data/schema coupling and match/serde break risk |
| 29 | impl | std::fmt::Display for PumpMode | method behavior and trait semantics |
| 30 | fn | fmt | API and behavior coupling to callers/implementers |
| 41 | struct | HydActuator | data/schema coupling and match/serde break risk |
| 55 | impl | HydActuator | method behavior and trait semantics |
| 56 | fn | new | API and behavior coupling to callers/implementers |
| 73 | fn | area_cap_cm2 | API and behavior coupling to callers/implementers |
| 76 | fn | area_rod_cm2 | API and behavior coupling to callers/implementers |
| 81 | fn | update | API and behavior coupling to callers/implementers |
| 137 | fn | position_pct | API and behavior coupling to callers/implementers |
| 144 | struct | EcuHcm | data/schema coupling and match/serde break risk |
| 198 | impl | Default for EcuHcm | method behavior and trait semantics |
| 199 | fn | default | API and behavior coupling to callers/implementers |
| 204 | impl | EcuHcm | method behavior and trait semantics |
| 205 | fn | new | API and behavior coupling to callers/implementers |
| 244 | fn | tick | API and behavior coupling to callers/implementers |
| 385 | fn | set_hitch_cmd | API and behavior coupling to callers/implementers |
| 388 | fn | set_loader_lift_cmd | API and behavior coupling to callers/implementers |
| 391 | fn | set_loader_tilt_cmd | API and behavior coupling to callers/implementers |
| 394 | fn | set_aux_cmd | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Hydraulic Control Module (HCM) — SA 0x1E. | comment/documentation; changing affects understanding and intent t
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Controls the entire hydraulic system on a heavy machine: | comment/documentation; changing affects understandi
| 4 | //!   • Variable-displacement axial-piston pump (load-sensing) | comment/documentation; changing affects understan
| 5 | //!   • Main system pressure regulation via PRV | comment/documentation; changing affects understanding and intent
| 6 | //!   • 3-point hitch electro-hydraulic cylinder control | comment/documentation; changing affects understanding a
| 7 | //!   • PTO wet clutch hydraulic engagement | comment/documentation; changing affects understanding and intent tra
| 8 | //!   • 4× remote auxiliary spool valve banks | comment/documentation; changing affects understanding and intent t
| 9 | //!   • Brake charge accumulator | comment/documentation; changing affects understanding and intent traceability.
| 10 | //!   • Hydraulic-to-oil cooling circuit | comment/documentation; changing affects understanding and intent trace
| 11 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 12 | //! J1939 SA: 0x1E. Transmits proprietary status heartbeat (100 ms). | comment/documentation; changing affects un
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 | use crate::j1939::{addr, pgn, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | pub const HCM_SA: u8 = addr::HITCH; // 0x1E | support logic; impact depends on neighboring block and data flow.
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | // ■■ Pump Mode ■■■
```

```
| 42 |     pub name: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 43 |     pub bore_mm: f64,       // cylinder bore | support logic; impact depends on neighboring block and data flow.
| 44 |     pub rod_mm: f64,        // rod diameter | support logic; impact depends on neighboring block and data flow.
| 45 |     pub stroke_mm: f64,     // full stroke | support logic; impact depends on neighboring block and data flow. |
| 46 |     pub position_mm: f64,   // current extension (0=fully retracted) | support logic; impact depends on neighbori
| 47 |     pub velocity_mm_s: f64, // positive = extending | support logic; impact depends on neighboring block and data
| 48 |     pub pres_a_bar: f64,    // cap-end pressure | support logic; impact depends on neighboring block and data flo
| 49 |     pub pres_b_bar: f64,    // rod-end pressure | support logic; impact depends on neighboring block and data flo
| 50 |     pub force_kn: f64,      // net push force (positive = extending) | support logic; impact depends on neighbori
| 51 |     pub valve_cmd: f64,     // −1 retract, 0 hold, +1 extend | support logic; impact depends on neighboring block
| 52 |     pub leakage_cc_m: f64,  // internal leakage cm³/min | support logic; impact depends on neighboring block and
| 53 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 | (blank) | blank separator; no runtime behavior. |
| 55 | impl HydActuator { | implementation binding; behavior semantics and trait conformance live here. |
| 56 |     pub fn new(name: &'static str, bore_mm: f64, rod_mm: f64, stroke_mm: f64) -> Self { | function signature/body
| 57 |         HydActuator { | support logic; impact depends on neighboring block and data flow. |
| 58 |             name, | support logic; impact depends on neighboring block and data flow. |
| 59 |             bore_mm, | support logic; impact depends on neighboring block and data flow. |
| 60 |             rod_mm, | support logic; impact depends on neighboring block and data flow. |
| 61 |             stroke_mm, | support logic; impact depends on neighboring block and data flow. |
| 62 |             position_mm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 63 |             velocity_mm_s: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 64 |             pres_a_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 65 |             pres_b_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 66 |             force_kn: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 67 |             valve_cmd: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 68 |             leakage_cc_m: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 69 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 70 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 | (blank) | blank separator; no runtime behavior. |
| 72 |     /// Area of each side in cm² | comment/documentation; changing affects understanding and intent traceability
| 73 |     pub fn area_cap_cm2(&self) -> f64 { | function signature/body; changes affect API behavior and control-flow.
| 74 |         std::f64::consts::PI * (self.bore_mm / 20.0).powi(2) | support logic; impact depends on neighboring bloc
| 75 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 76 |     pub fn area_rod_cm2(&self) -> f64 { | function signature/body; changes affect API behavior and control-flow.
| 77 |         self.area_cap_cm2() - std::f64::consts::PI * (self.rod_mm / 20.0).powi(2) | support logic; impact depends
| 78 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 79 | (blank) | blank separator; no runtime behavior. |
| 80 |     /// Update given supply pressure and flow from the pump. Returns flow consumed (L/min). | comment/documentati
| 81 |     pub fn update(&mut self, supply_bar: f64, dt: f64) -> f64 { | function signature/body; changes affect API bel
| 82 |         if self.valve_cmd.abs() < 0.01 { | runtime gate; condition changes can enable/disable critical paths. |
| 83 |             // Valve closed: hold position, pressures bleed through leakage | comment/documentation; changing af
| 84 |             self.pres_a_bar = (self.pres_a_bar - 2.0 * dt).max(0.0); | support logic; impact depends on neighbori
| 85 |             self.pres_b_bar = (self.pres_b_bar - 2.0 * dt).max(0.0); | support logic; impact depends on neighbori
| 86 |             self.velocity_mm_s = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 87 |             return self.leakage_cc_m / 1000.0; | support logic; impact depends on neighboring block and data flow
| 88 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 | (blank) | blank separator; no runtime behavior. |
| 90 |         let cmd = self.valve_cmd.clamp(-1.0, 1.0); | support logic; impact depends on neighboring block and data
| 91 |         let extending = cmd > 0.0; | support logic; impact depends on neighboring block and data flow. |
| 92 | (blank) | blank separator; no runtime behavior. |
| 93 |         // Pressure at active side ≈ supply × valve-opening factor | comment/documentation; changing affects unde
| 94 |         if extending { | runtime gate; condition changes can enable/disable critical paths. |
| 95 |             self.pres_a_bar = supply_bar * cmd.abs() * 0.92; | support logic; impact depends on neighboring bloc
| 96 |             self.pres_b_bar = | support logic; impact depends on neighboring block and data flow. |
| 97 |                 (self.pres_a_bar * self.area_cap_cm2() / self.area_rod_cm2().max(0.01)) * 0.15; | support logic;
| 98 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 99 |             self.pres_b_bar = supply_bar * cmd.abs() * 0.92; | support logic; impact depends on neighboring bloc
| 100 |             self.pres_a_bar = self.pres_b_bar * 0.1; | support logic; impact depends on neighboring block and da
| 101 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 102 | (blank) | blank separator; no runtime behavior. |
| 103 |         // Net force | comment/documentation; changing affects understanding and intent traceability. |
| 104 |         let f_cap = self.pres_a_bar * self.area_cap_cm2() * 10.0; // N | support logic; impact depends on neigh
| 105 |         let f_rod = self.pres_b_bar * self.area_rod_cm2() * 10.0; | support logic; impact depends on neighborin
| 106 |         self.force_kn = (f_cap - f_rod) / 1000.0; | support logic; impact depends on neighboring block and data
| 107 | (blank) | blank separator; no runtime behavior. |
| 108 |         // Velocity from flow-through valve (orifice model Q = Cv × cmd × √ΔP) | comment/documentation; changing
| 109 |         let dp = (supply_bar | support logic; impact depends on neighboring block and data flow. |
| 110 |             - if extending { | support logic; impact depends on neighboring block and data flow. |
| 111 |                 self.pres_b_bar | support logic; impact depends on neighboring block and data flow. |
| 112 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 113 |                 self.pres_a_bar | support logic; impact depends on neighboring block and data flow. |
| 114 |             }) | support logic; impact depends on neighboring block and data flow. |
| 115 |         .max(0.0); | support logic; impact depends on neighboring block and data flow. |
| 116 |         let flow_lpm = 40.0 * cmd.abs() * dp.sqrt() / 10.0; // simplified Cv | support logic; impact depends on
| 117 |         let area_cm2 = if extending { | support logic; impact depends on neighboring block and data flow. |
```

```
| 118 |             self.area_cap_cm2() | support logic; impact depends on neighboring block and data flow. |
| 119 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 120 |             self.area_rod_cm2() | support logic; impact depends on neighboring block and data flow. |
| 121 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 122 |         self.velocity_mm_s = flow_lpm * 1000.0 / 60.0 / area_cm2 * cmd.signum(); | support logic; impact depends
| 123 | (blank) | blank separator; no runtime behavior. |
| 124 |         // Integrate position — hard stops at 0 and stroke | comment/documentation; changing affects understandi
| 125 |         self.position_mm = (self.position_mm + self.velocity_mm_s * dt).clamp(0.0, self.stroke_mm); | support lo
| 126 | (blank) | blank separator; no runtime behavior. |
| 127 |         // Stop at end-of-stroke | comment/documentation; changing affects understanding and intent traceability
| 128 |         if (self.position_mm <= 0.0 && self.velocity_mm_s < 0.0) | runtime gate; condition changes can enable/di
| 129 |             \|\| (self.position_mm >= self.stroke_mm && self.velocity_mm_s > 0.0) | support logic; impact depend
| 130 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 131 |             self.velocity_mm_s = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 132 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 133 | (blank) | blank separator; no runtime behavior. |
| 134 |         flow_lpm.abs() | support logic; impact depends on neighboring block and data flow. |
| 135 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 136 | (blank) | blank separator; no runtime behavior. |
| 137 |     pub fn position_pct(&self) -> f64 { | function signature/body; changes affect API behavior and control-flow
| 138 |         self.position_mm / self.stroke_mm.max(1.0) * 100.0 | support logic; impact depends on neighboring block
| 139 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 140 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 141 | (blank) | blank separator; no runtime behavior. |
| 142 | // ■■ HCM ■■■
```

```
| 194 |     t_hb: f64, | support logic; impact depends on neighboring block and data flow. |
| 195 |     sim_time_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 196 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 197 | (blank) | blank separator; no runtime behavior. |
| 198 | impl Default for EcuHcm { | implementation binding; behavior semantics and trait conformance live here. |
| 199 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 200 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 201 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 202 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 203 | (blank) | blank separator; no runtime behavior. |
| 204 | impl EcuHcm { | implementation binding; behavior semantics and trait conformance live here. |
| 205 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 206 |         EcuHcm { | support logic; impact depends on neighboring block and data flow. |
| 207 |             sa: HCM_SA, | support logic; impact depends on neighboring block and data flow. |
| 208 |             pump_displacement_cc: 100.0, // 100 cc/rev (large agricultural pump) | support logic; impact depends
| 209 |             pump_actual_cc: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 210 |             pump_mode: PumpMode::Standby, | support logic; impact depends on neighboring block and data flow. |
| 211 |             pump_rpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 212 |             pump_flow_lpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 213 |             pump_efficiency: 0.92, | support logic; impact depends on neighboring block and data flow. |
| 214 |             system_pressure_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 215 |             relief_pressure_bar: 230.0, // 230 bar PRV | support logic; impact depends on neighboring block and
| 216 |             ls_signal_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 217 |             charge_pres_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 218 |             return_pres_bar: 5.0, | support logic; impact depends on neighboring block and data flow. |
| 219 |             pilot_pres_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 220 |             fluid_temp_c: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 221 |             fluid_level_pct: 92.0, | support logic; impact depends on neighboring block and data flow. |
| 222 |             filter_dp_bar: 0.8, | support logic; impact depends on neighboring block and data flow. |
| 223 |             filter_bypass_open: false, | support logic; impact depends on neighboring block and data flow. |
| 224 |             hitch_cylinder: HydActuator::new("HITCH", 100.0, 50.0, 400.0), | support logic; impact depends on n
| 225 |             loader_lift: HydActuator::new("LOADER-LIFT", 80.0, 40.0, 800.0), | support logic; impact depends on
| 226 |             loader_tilt: HydActuator::new("LOADER-TILT", 70.0, 35.0, 300.0), | support logic; impact depends on
| 227 |             aux_valve: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 228 |             aux_flow_lpm: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 229 |             aux_pres_bar: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 230 |             brake_acc_pres_bar: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 231 |             brake_acc_charged: false, | support logic; impact depends on neighboring block and data flow. |
| 232 |             alarm_high_temp: false, | support logic; impact depends on neighboring block and data flow. |
| 233 |             alarm_low_level: false, | support logic; impact depends on neighboring block and data flow. |
| 234 |             alarm_filter: false, | support logic; impact depends on neighboring block and data flow. |
| 235 |             alarm_low_pressure: false, | support logic; impact depends on neighboring block and data flow. |
| 236 |             hydraulic_power_kw: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 237 |             total_flow_demand_lpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 238 |             t_hb: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 239 |             sim_time_s: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 240 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 241 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 242 | (blank) | blank separator; no runtime behavior. |
| 243 |     /// Main HCM tick. `engine_rpm` from ECM, `hitch_cmd` from implement controller (−1..+1). | comment/document
| 244 |     pub fn tick( | function signature/body; changes affect API behavior and control-flow. |
| 245 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 246 |         engine_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 247 |         hitch_cmd: f64, | support logic; impact depends on neighboring block and data flow. |
| 248 |         loader_lift_cmd: f64, | support logic; impact depends on neighboring block and data flow. |
| 249 |         loader_tilt_cmd: f64, | support logic; impact depends on neighboring block and data flow. |
| 250 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 251 |     ) -> Vec<J1939Frame> { | support logic; impact depends on neighboring block and data flow. |
| 252 |         self.sim_time_s += dt; | support logic; impact depends on neighboring block and data flow. |
| 253 |         self.pump_rpm = engine_rpm * 1.12; // pump driven at 1.12× engine (PTO shaft) | support logic; impact de
| 254 | (blank) | blank separator; no runtime behavior. |
| 255 |         // ■ Determine load-sensing demand ■■■
```

```
| 270 |         for &cmd in self.aux_valve.iter() { | iteration logic; may alter timing, throughput, and safety constrai
| 271 |             if cmd.abs() > 0.01 { | runtime gate; condition changes can enable/disable critical paths. |
| 272 |                 self.ls_signal_bar = self.ls_signal_bar.max(100.0 + cmd.abs() * 100.0); | support logic; impact
| 273 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 274 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 275 | (blank) | blank separator; no runtime behavior. |
| 276 |         // ■ Load-sensing pump control ■■■
```

```
| 346 |         let heat_kw = (kw_in - kw_useful).max(0.0); | support logic; impact depends on neighboring block and dat
| 347 |         self.hydraulic_power_kw = kw_useful; | support logic; impact depends on neighboring block and data flow
| 348 |         let tgt_temp = 45.0 + heat_kw * 1.5; | support logic; impact depends on neighboring block and data flow
| 349 |         self.fluid_temp_c += (tgt_temp - self.fluid_temp_c) * dt * 0.008; | support logic; impact depends on nei
| 350 | (blank) | blank separator; no runtime behavior. |
| 351 |         // ■ Filter ■■■
```

```
| start_self_test | 1 |
| tick | 1 |
| build_id | 1 |
| push | 1 |
| from_raw | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | const | ICM_SA | namespace and compile-time linkage |
| 12 | struct | Gauge | data/schema coupling and match/serde break risk |
| 25 | impl | Gauge | method behavior and trait semantics |
| 26 | fn | new | API and behavior coupling to callers/implementers |
| 40 | fn | with_warn_hi | API and behavior coupling to callers/implementers |
| 45 | fn | with_warn_lo | API and behavior coupling to callers/implementers |
| 50 | fn | update | API and behavior coupling to callers/implementers |
| 58 | fn | fraction | API and behavior coupling to callers/implementers |
| 65 | enum | LampColor | data/schema coupling and match/serde break risk |
| 74 | struct | WarningLamp | data/schema coupling and match/serde break risk |
| 80 | impl | WarningLamp | method behavior and trait semantics |
| 81 | fn | new | API and behavior coupling to callers/implementers |
| 88 | fn | set | API and behavior coupling to callers/implementers |
| 95 | struct | TripComputer | data/schema coupling and match/serde break risk |
| 105 | impl | Default for TripComputer | method behavior and trait semantics |
| 106 | fn | default | API and behavior coupling to callers/implementers |
| 111 | impl | TripComputer | method behavior and trait semantics |
| 112 | fn | new | API and behavior coupling to callers/implementers |
| 124 | fn | update | API and behavior coupling to callers/implementers |
| 136 | fn | reset | API and behavior coupling to callers/implementers |
| 143 | struct | EcuIcm | data/schema coupling and match/serde break risk |
| 192 | impl | Default for EcuIcm | method behavior and trait semantics |
| 193 | fn | default | API and behavior coupling to callers/implementers |
| 198 | impl | EcuIcm | method behavior and trait semantics |
| 199 | fn | new | API and behavior coupling to callers/implementers |
| 254 | fn | start_self_test | API and behavior coupling to callers/implementers |
| 263 | fn | tick | API and behavior coupling to callers/implementers |
| 397 | fn | update_lamps | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Instrument Cluster Module (ICM/DASH) — SA 0x1C. | comment/documentation; changing affects understanding and in
| 2 | //! Receives J1939 data from all ECUs, presents live gauge readings. | comment/documentation; changing affects und
| 3 | //! Tracks odometer, trip computer, warning lamp states. | comment/documentation; changing affects understanding a
| 4 | //! J1939: listens only (display module); sends heartbeat on Prop-B (100ms). | comment/documentation; changing aff
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use crate::j1939::{self, addr, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | pub const ICM_SA: u8 = addr::INSTRUMENT; // 0x1C | support logic; impact depends on neighboring block and data flo
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | // ■■ Gauge ■■■
```

```
| 33 |             warn_lo: None, | support logic; impact depends on neighboring block and data flow. |
| 34 |             warn_hi: None, | support logic; impact depends on neighboring block and data flow. |
| 35 |             crit_hi: None, | support logic; impact depends on neighboring block and data flow. |
| 36 |             in_warning: false, | support logic; impact depends on neighboring block and data flow. |
| 37 |             in_critical: false, | support logic; impact depends on neighboring block and data flow. |
| 38 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 40 |     pub fn with_warn_hi(mut self, w: f64, c: f64) -> Self { | function signature/body; changes affect API behavio
| 41 |         self.warn_hi = Some(w); | support logic; impact depends on neighboring block and data flow. |
| 42 |         self.crit_hi = Some(c); | support logic; impact depends on neighboring block and data flow. |
| 43 |         self | support logic; impact depends on neighboring block and data flow. |
| 44 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 45 |     pub fn with_warn_lo(mut self, w: f64) -> Self { | function signature/body; changes affect API behavior and co
| 46 |         self.warn_lo = Some(w); | support logic; impact depends on neighboring block and data flow. |
| 47 |         self | support logic; impact depends on neighboring block and data flow. |
| 48 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 |     pub fn update(&mut self, val: f64) { | function signature/body; changes affect API behavior and control-flow
| 51 |         self.value = val.clamp(self.min, self.max); | support logic; impact depends on neighboring block and data
| 52 |         self.in_warning = self.warn_hi.is_some_and(\|w\| self.value > w) | support logic; impact depends on neigh
| 53 |             \|\| self.warn_lo.is_some_and(\|l\| self.value < l); | support logic; impact depends on neighboring b
| 54 |         self.in_critical = self.crit_hi.is_some_and(\|c\| self.value > c); | support logic; impact depends on nei
| 55 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 56 | (blank) | blank separator; no runtime behavior. |
| 57 |     /// Fraction 0-1 of gauge arc | comment/documentation; changing affects understanding and intent traceability
| 58 |     pub fn fraction(&self) -> f64 { | function signature/body; changes affect API behavior and control-flow. |
| 59 |         (self.value - self.min) / (self.max - self.min) | support logic; impact depends on neighboring block and
| 60 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 61 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 62 | (blank) | blank separator; no runtime behavior. |
| 63 | // ■■ Warning Lamp ■■■
```

```
| 109 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 110 | (blank) | blank separator; no runtime behavior. |
| 111 | impl TripComputer { | implementation binding; behavior semantics and trait conformance live here. |
| 112 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 113 |         TripComputer { | support logic; impact depends on neighboring block and data flow. |
| 114 |             trip_distance_km: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 115 |             trip_fuel_l: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 116 |             trip_hours: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 117 |             avg_fuel_lph: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 118 |             avg_speed_kmh: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 119 |             samples: 0, | support logic; impact depends on neighboring block and data flow. |
| 120 |             speed_sum: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 121 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 122 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 123 | (blank) | blank separator; no runtime behavior. |
| 124 |     pub fn update(&mut self, speed_kmh: f64, fuel_lph: f64, dt: f64) { | function signature/body; changes affec
| 125 |         self.trip_distance_km += speed_kmh * dt / 3600.0; | support logic; impact depends on neighboring block a
| 126 |         self.trip_fuel_l += fuel_lph * dt / 3600.0; | support logic; impact depends on neighboring block and da
| 127 |         self.trip_hours += dt / 3600.0; | support logic; impact depends on neighboring block and data flow. |
| 128 |         self.speed_sum += speed_kmh; | support logic; impact depends on neighboring block and data flow. |
| 129 |         self.samples += 1; | support logic; impact depends on neighboring block and data flow. |
| 130 |         self.avg_speed_kmh = self.speed_sum / self.samples.max(1) as f64; | support logic; impact depends on ne
| 131 |         if self.trip_hours > 0.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 132 |             self.avg_fuel_lph = self.trip_fuel_l / self.trip_hours; | support logic; impact depends on neighbor
| 133 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 134 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 135 | (blank) | blank separator; no runtime behavior. |
| 136 |     pub fn reset(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 137 |         *self = Self::new(); | support logic; impact depends on neighboring block and data flow. |
| 138 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 139 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 140 | (blank) | blank separator; no runtime behavior. |
| 141 | // ■■ ICM ■■■
```

```
| 185 |     pub self_test_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 186 |     self_test_timer: f64, | support logic; impact depends on neighboring block and data flow. |
| 187 | (blank) | blank separator; no runtime behavior. |
| 188 |     // ■ TX timer ■■■
```

```
| 261 | (blank) | blank separator; no runtime behavior. |
| 262 |     /// Main ICM tick — processes received CAN frames, updates gauges. | comment/documentation; changing affects
| 263 |     pub fn tick(&mut self, received: &[J1939Frame], dt: f64) -> Vec<J1939Frame> { | function signature/body; cha
| 264 |         // ■ Self-test ■■■
```

```
| 337 |                         if v.spn == 247 { | runtime gate; condition changes can enable/disable critical paths.
| 338 |                             self.engine_hours = v.physical; | support logic; impact depends on neighboring bloc
| 339 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 340 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 341 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 342 |                 p if p == pgn::CCVS => { | support logic; impact depends on neighboring block and data flow. |
| 343 |                     for v in &frame.decoded { | iteration logic; may alter timing, throughput, and safety const
| 344 |                         if v.spn == 84 { | runtime gate; condition changes can enable/disable critical paths. |
| 345 |                             self.vehicle_speed = v.physical; | support logic; impact depends on neighboring bloc
| 346 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 347 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 348 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 349 |                 p if p == pgn::DM1 => { | support logic; impact depends on neighboring block and data flow. |
| 350 |                     // Read lamp bits from byte 0 | comment/documentation; changing affects understanding and i
| 351 |                     let b0 = frame.data[0]; | support logic; impact depends on neighboring block and data flow.
| 352 |                     self.mil_active = (b0 & 0x40) != 0; | support logic; impact depends on neighboring block an
| 353 |                     self.red_active = (b0 & 0x10) != 0; | support logic; impact depends on neighboring block an
| 354 |                     self.amber_active = (b0 & 0x04) != 0; | support logic; impact depends on neighboring block a
| 355 |                     self.protect_active = (b0 & 0x01) != 0; | support logic; impact depends on neighboring bloc
| 356 |                     self.active_dtc_count = if frame.data[2] != 0xFF { 1 } else { 0 }; | support logic; impact
| 357 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 358 |                 _ => {} | support logic; impact depends on neighboring block and data flow. |
| 359 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 360 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 361 | (blank) | blank separator; no runtime behavior. |
| 362 |         // ■ Update all gauges ■■■
```

```
| 413 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 414 |                 "MIL" => { | support logic; impact depends on neighboring block and data flow. |
| 415 |                     if self.mil_active { | runtime gate; condition changes can enable/disable critical paths. |
| 416 |                         LampColor::Yellow | support logic; impact depends on neighboring block and data flow. |
| 417 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 418 |                         LampColor::Off | support logic; impact depends on neighboring block and data flow. |
| 419 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 420 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 421 |                 "PROTECT" => { | support logic; impact depends on neighboring block and data flow. |
| 422 |                     if self.protect_active { | runtime gate; condition changes can enable/disable critical paths
| 423 |                         LampColor::Yellow | support logic; impact depends on neighboring block and data flow. |
| 424 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 425 |                         LampColor::Off | support logic; impact depends on neighboring block and data flow. |
| 426 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 427 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 428 |                 "OIL_PRESS" => { | support logic; impact depends on neighboring block and data flow. |
| 429 |                     if self.oil_pressure_kpa < 100.0 && self.engine_rpm > 500.0 { | runtime gate; condition char
| 430 |                         LampColor::Red | support logic; impact depends on neighboring block and data flow. |
| 431 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 432 |                         LampColor::Off | support logic; impact depends on neighboring block and data flow. |
| 433 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 434 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 435 |                 "COOLANT_T" => { | support logic; impact depends on neighboring block and data flow. |
| 436 |                     if self.coolant_temp > 105.0 { | runtime gate; condition changes can enable/disable critical
| 437 |                         LampColor::Red | support logic; impact depends on neighboring block and data flow. |
| 438 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 439 |                         LampColor::Off | support logic; impact depends on neighboring block and data flow. |
| 440 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 441 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 442 |                 "DEF_LOW" => { | support logic; impact depends on neighboring block and data flow. |
| 443 |                     if self.def_level_pct < 10.0 { | runtime gate; condition changes can enable/disable critical
| 444 |                         LampColor::Yellow | support logic; impact depends on neighboring block and data flow. |
| 445 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 446 |                         LampColor::Off | support logic; impact depends on neighboring block and data flow. |
| 447 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 448 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 449 |                 "FUEL_LOW" => { | support logic; impact depends on neighboring block and data flow. |
| 450 |                     if self.fuel_level_pct < 10.0 { | runtime gate; condition changes can enable/disable critica
| 451 |                         LampColor::Yellow | support logic; impact depends on neighboring block and data flow. |
| 452 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 453 |                         LampColor::Off | support logic; impact depends on neighboring block and data flow. |
| 454 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 455 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 456 |                 "CHARGE" => { | support logic; impact depends on neighboring block and data flow. |
| 457 |                     if self.battery_volts < 12.0 { | runtime gate; condition changes can enable/disable critical
| 458 |                         LampColor::Yellow | support logic; impact depends on neighboring block and data flow. |
| 459 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 460 |                         LampColor::Off | support logic; impact depends on neighboring block and data flow. |
| 461 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 462 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 463 |                 _ => LampColor::Off, | support logic; impact depends on neighboring block and data flow. |
| 464 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 465 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 466 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 467 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/ecu_tcm.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 716 | Non-empty: 647 | Symbol count: 43

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| fmt | 6 |
| request_downshift | 6 |
| begin_shift | 6 |
| new | 5 |
| tick | 3 |
| request_upshift | 3 |
| update_ivt | 2 |
| update_clutch | 2 |
```

```
| auto_shift_logic | 2 |
| update_shift | 2 |
| compute_gear_label | 2 |
| push | 2 |
| build_etc1 | 2 |
| build_etc2 | 2 |
| from_raw | 2 |
| build_id | 2 |
| default | 1 |
| manual_upshift | 1 |
| manual_downshift | 1 |
| set_direction | 1 |
| set_neutral | 1 |
| toggle_creeper | 1 |
| toggle_auto | 1 |
| d4_partial_throttle_does_not_pin_to_50_kmh | 1 |
| d4_full_throttle_reaches_transport_band_without_exceeding_limit | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 16 | const | TCM_SA | namespace and compile-time linkage |
| 20 | enum | TransmissionType | data/schema coupling and match/serde break risk |
| 30 | enum | GearRange | data/schema coupling and match/serde break risk |
| 41 | impl | std::fmt::Display for GearRange | method behavior and trait semantics |
| 42 | fn | fmt | API and behavior coupling to callers/implementers |
| 55 | enum | Direction | data/schema coupling and match/serde break risk |
| 62 | impl | std::fmt::Display for Direction | method behavior and trait semantics |
| 63 | fn | fmt | API and behavior coupling to callers/implementers |
| 75 | enum | ClutchState | data/schema coupling and match/serde break risk |
| 86 | impl | std::fmt::Display for ClutchState | method behavior and trait semantics |
| 87 | fn | fmt | API and behavior coupling to callers/implementers |
| 99 | enum | ShiftQuality | data/schema coupling and match/serde break risk |
| 105 | impl | std::fmt::Display for ShiftQuality | method behavior and trait semantics |
| 106 | fn | fmt | API and behavior coupling to callers/implementers |
| 118 | enum | AutoShiftMode | data/schema coupling and match/serde break risk |
| 130 | const | GEAR_RATIOS | namespace and compile-time linkage |
| 141 | const | TIRE_RADIUS_M | namespace and compile-time linkage |
| 142 | const | MAX_SPEED_KMH | namespace and compile-time linkage |
| 146 | struct | EcuTcm | data/schema coupling and match/serde break risk |
| 211 | impl | Default for EcuTcm | method behavior and trait semantics |
| 212 | fn | default | API and behavior coupling to callers/implementers |
| 217 | impl | EcuTcm | method behavior and trait semantics |
| 218 | fn | new | API and behavior coupling to callers/implementers |
| 263 | fn | tick | API and behavior coupling to callers/implementers |
| 383 | fn | update_clutch | API and behavior coupling to callers/implementers |
| 424 | fn | auto_shift_logic | API and behavior coupling to callers/implementers |
| 479 | fn | request_upshift | API and behavior coupling to callers/implementers |
| 496 | fn | request_downshift | API and behavior coupling to callers/implementers |
| 513 | fn | begin_shift | API and behavior coupling to callers/implementers |
| 535 | fn | update_shift | API and behavior coupling to callers/implementers |
| 572 | fn | update_ivt | API and behavior coupling to callers/implementers |
| 590 | fn | manual_upshift | API and behavior coupling to callers/implementers |
| 595 | fn | manual_downshift | API and behavior coupling to callers/implementers |
| 600 | fn | set_direction | API and behavior coupling to callers/implementers |
| 603 | fn | set_neutral | API and behavior coupling to callers/implementers |
| 607 | fn | toggle_creeper | API and behavior coupling to callers/implementers |
| 610 | fn | toggle_auto | API and behavior coupling to callers/implementers |
| 618 | fn | compute_gear_label | API and behavior coupling to callers/implementers |
| 628 | fn | build_etc1 | API and behavior coupling to callers/implementers |
| 654 | fn | build_etc2 | API and behavior coupling to callers/implementers |
| 677 | module | tests | namespace and compile-time linkage |
| 681 | fn | d4_partial_throttle_does_not_pin_to_50_kmh | API and behavior coupling to callers/implementers |
| 701 | fn | d4_full_throttle_reaches_transport_band_without_exceeding_limit | API and behavior coupling to callers/impl
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Transmission Control Module (TCM) — Powershift 16F/16R + optional IVT/CVT. | comment/documentation; changing a
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Models a real agricultural powershift transmission: | comment/documentation; changing affects understanding an
| 4 | //!   • 4 ranges (A=crawl, B=field-low, C=field-high, D=road) | comment/documentation; changing affects understand
| 5 | //!   • 4 powershift speeds per range = 16 forward / 16 reverse | comment/documentation; changing affects understa
```

```
| 6 | //!   • Wet multi-disc powershift clutches (P1–P4) with fill, modulate, lock phases | comment/documentation; chang
| 7 | //!   • Electronic shuttle (direction change under load) | comment/documentation; changing affects understanding a
| 8 | //!   • Auto-shift based on engine load and RPM | comment/documentation; changing affects understanding and intent
| 9 | //!   • Creeper gear for slow tillage work | comment/documentation; changing affects understanding and intent trac
| 10 | //!   • Ground Speed Management (GSM): maintains target speed regardless of load | comment/documentation; changin
| 11 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 12 | //! J1939 SA: 0x03, Transmits ETC1 (20ms), ETC2 (20ms). | comment/documentation; changing affects understanding a
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 | use crate::j1939::{self, addr, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | pub const TCM_SA: u8 = addr::TRANSMISSION; // 0x03 | support logic; impact depends on neighboring block and data
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | // ■■ Transmission Type ■■■
```

```
| 82 |     /// Disengaged — no torque transfer | comment/documentation; changing affects understanding and intent tracea
| 83 |     Open, | support logic; impact depends on neighboring block and data flow. |
| 84 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 85 | (blank) | blank separator; no runtime behavior. |
| 86 | impl std::fmt::Display for ClutchState { | implementation binding; behavior semantics and trait conformance live
| 87 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 88 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 89 |             ClutchState::Filling => write!(f, "FILLING "), | protocol or I/O critical; changes may impact externa
| 90 |             ClutchState::Modulating => write!(f, "MODULATG"), | protocol or I/O critical; changes may impact exte
| 91 |             ClutchState::Locked => write!(f, "LOCKED  "), | protocol or I/O critical; changes may impact external
| 92 |             ClutchState::Open => write!(f, "OPEN    "), | protocol or I/O critical; changes may impact external i
| 93 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 94 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 95 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 96 | (blank) | blank separator; no runtime behavior. |
| 97 | // ■■ Shift Quality ■■■
```

```
| 158 |     pub output_torque_nm: f64, | support logic; impact depends on neighboring block and data flow. |
| 159 |     pub ground_speed_kmh: f64, | support logic; impact depends on neighboring block and data flow. |
| 160 |     pub gear_ratio: f64, // current overall ratio | support logic; impact depends on neighboring block and data
| 161 | (blank) | blank separator; no runtime behavior. |
| 162 |     // ■ Clutch pack ■■■
```

```
| 234 |             clutch_phase_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 235 |             is_shifting: false, | support logic; impact depends on neighboring block and data flow. |
| 236 |             last_shift_qual: ShiftQuality::Normal, | support logic; impact depends on neighboring block and data
| 237 |             total_shifts: 0, | support logic; impact depends on neighboring block and data flow. |
| 238 |             pending_range: None, | support logic; impact depends on neighboring block and data flow. |
| 239 |             pending_speed: None, | support logic; impact depends on neighboring block and data flow. |
| 240 |             pending_dir: None, | support logic; impact depends on neighboring block and data flow. |
| 241 |             shift_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 242 |             auto_shift_cooldown_s: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 243 |             shift_duration: 0.30, | support logic; impact depends on neighboring block and data flow. |
| 244 |             auto_mode: AutoShiftMode::Auto, | support logic; impact depends on neighboring block and data flow.
| 245 |             gsm_target_kmh: 8.0, | support logic; impact depends on neighboring block and data flow. |
| 246 |             gsm_enabled: false, | support logic; impact depends on neighboring block and data flow. |
| 247 |             ivt_ratio: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 248 |             creeper_engaged: false, | support logic; impact depends on neighboring block and data flow. |
| 249 |             creeper_ratio: 4.0, | support logic; impact depends on neighboring block and data flow. |
| 250 |             oil_temp_c: 30.0, | support logic; impact depends on neighboring block and data flow. |
| 251 |             sump_temp_c: 30.0, | support logic; impact depends on neighboring block and data flow. |
| 252 |             is_neutral: true, | support logic; impact depends on neighboring block and data flow. |
| 253 |             handbrake_interlock: false, | support logic; impact depends on neighboring block and data flow. |
| 254 |             t_etc1: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 255 |             t_etc2: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 256 |             total_distance_km: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 257 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 258 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 259 | (blank) | blank separator; no runtime behavior. |
| 260 |     // ■■■
```

```
| 310 |             ClutchState::Open => 0.0, | support logic; impact depends on neighboring block and data flow. |
| 311 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 312 |         target_speed_kmh *= coupling; | support logic; impact depends on neighboring block and data flow. |
| 313 | (blank) | blank separator; no runtime behavior. |
| 314 |         let accel_limit_kmh_s = 1.4 + (throttle_pct / 100.0) * 4.6; | support logic; impact depends on neighbori
| 315 |         let decel_limit_kmh_s = 2.0 + (brake_pct / 100.0) * 14.0; | support logic; impact depends on neighboring
| 316 |         let max_rise = accel_limit_kmh_s * dt; | support logic; impact depends on neighboring block and data flo
| 317 |         let max_drop = decel_limit_kmh_s * dt; | support logic; impact depends on neighboring block and data flo
| 318 |         let delta = target_speed_kmh - self.ground_speed_kmh; | support logic; impact depends on neighboring blo
| 319 |         if delta >= 0.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 320 |             self.ground_speed_kmh += delta.min(max_rise); | support logic; impact depends on neighboring block a
| 321 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 322 |             self.ground_speed_kmh += delta.max(-max_drop); | support logic; impact depends on neighboring block
| 323 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 324 |         self.ground_speed_kmh = self.ground_speed_kmh.clamp(0.0, MAX_SPEED_KMH); | support logic; impact depends
| 325 | (blank) | blank separator; no runtime behavior. |
| 326 |         // ■ Auto-shift logic ■■■
```

```
| 386 |                 self.clutch_slip_pct = 100.0; | support logic; impact depends on neighboring block and data flow
| 387 |                 self.clutch_fill_pres = 0.0; | support logic; impact depends on neighboring block and data flow
| 388 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 389 |             ClutchState::Filling => { | support logic; impact depends on neighboring block and data flow. |
| 390 |                 self.clutch_phase_timer += dt; | support logic; impact depends on neighboring block and data flo
| 391 |                 // Fill ramp 0 → 10 bar in 80-150 ms (faster when warm) | comment/documentation; changing affec
| 392 |                 let fill_time = (0.15 - self.oil_temp_c.min(80.0) / 80.0 * 0.07).max(0.05); | support logic; imp
| 393 |                 self.clutch_fill_pres = (self.clutch_phase_timer / fill_time * 10.0).min(10.0); | support logic
| 394 |                 if self.clutch_phase_timer >= fill_time { | runtime gate; condition changes can enable/disable c
| 395 |                     self.clutch_state = ClutchState::Modulating; | support logic; impact depends on neighboring
| 396 |                     self.clutch_phase_timer = 0.0; | support logic; impact depends on neighboring block and data
| 397 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 398 |                 self.clutch_slip_pct = 80.0; | support logic; impact depends on neighboring block and data flow
| 399 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 400 |             ClutchState::Modulating => { | support logic; impact depends on neighboring block and data flow. |
| 401 |                 self.clutch_phase_timer += dt; | support logic; impact depends on neighboring block and data flo
| 402 |                 // Modulate pressure 10 → 25 bar over ~250 ms, reducing slip | comment/documentation; changing
| 403 |                 let mod_time = 0.25; | support logic; impact depends on neighboring block and data flow. |
| 404 |                 let pres_ramp = self.clutch_phase_timer / mod_time; | support logic; impact depends on neighbori
| 405 |                 self.clutch_fill_pres = 10.0 + pres_ramp * 15.0; | support logic; impact depends on neighboring
| 406 |                 self.clutch_slip_pct = (80.0 * (1.0 - pres_ramp)).max(0.0); | support logic; impact depends on n
| 407 |                 if self.clutch_phase_timer >= mod_time { | runtime gate; condition changes can enable/disable cr
| 408 |                     self.clutch_state = ClutchState::Locked; | support logic; impact depends on neighboring bloc
| 409 |                     self.clutch_slip_pct = 0.0; | support logic; impact depends on neighboring block and data fl
| 410 |                     self.clutch_fill_pres = 25.0; | support logic; impact depends on neighboring block and data
| 411 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 412 |                 // Clutch temp from slip energy | comment/documentation; changing affects understanding and inte
| 413 |                 self.clutch_temp_c += self.clutch_slip_pct * torque_nm * 0.00005; | support logic; impact depend
| 414 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 415 |             ClutchState::Locked => { | support logic; impact depends on neighboring block and data flow. |
| 416 |                 self.clutch_fill_pres = 25.0; | support logic; impact depends on neighboring block and data flow
| 417 |                 self.clutch_slip_pct = 0.0; | support logic; impact depends on neighboring block and data flow.
| 418 |                 self.clutch_temp_c = (self.clutch_temp_c - 5.0 * dt).max(self.oil_temp_c); | support logic; impa
| 419 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 420 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 421 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 422 | (blank) | blank separator; no runtime behavior. |
| 423 |     // ■■■
```

```
| 462 |             return; | support logic; impact depends on neighboring block and data flow. |
| 463 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 464 | (blank) | blank separator; no runtime behavior. |
| 465 |         // Stall/lugging protection. | comment/documentation; changing affects understanding and intent traceabi
| 466 |         if engine_rpm < 820.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 467 |             self.request_downshift(); | support logic; impact depends on neighboring block and data flow. |
| 468 |             return; | support logic; impact depends on neighboring block and data flow. |
| 469 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 470 | (blank) | blank separator; no runtime behavior. |
| 471 |         if engine_rpm > upshift_rpm { | runtime gate; condition changes can enable/disable critical paths. |
| 472 |             self.request_upshift(); | support logic; impact depends on neighboring block and data flow. |
| 473 |         } else if engine_rpm < downshift_rpm { | support logic; impact depends on neighboring block and data flo
| 474 |             self.request_downshift(); | support logic; impact depends on neighboring block and data flow. |
| 475 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 476 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 477 | (blank) | blank separator; no runtime behavior. |
| 478 |     // ■■■
```

```
| 538 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 539 |         self.shift_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 540 |         // Mid-shift: engage new gear at ~50% of shift time | comment/documentation; changing affects understand
| 541 |         if self.shift_timer >= self.shift_duration * 0.5 { | runtime gate; condition changes can enable/disable
| 542 |             if let (Some(r), Some(s), Some(d)) = | runtime gate; condition changes can enable/disable critical |
| 543 |                 (self.pending_range, self.pending_speed, self.pending_dir) | support logic; impact depends on ne
| 544 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 545 |                 self.range = r; | support logic; impact depends on neighboring block and data flow. |
| 546 |                 self.speed_step = s.clamp(1, 4); | support logic; impact depends on neighboring block and data f
| 547 |                 self.direction = d; | support logic; impact depends on neighboring block and data flow. |
| 548 |                 self.pending_range = None; | support logic; impact depends on neighboring block and data flow.
| 549 |                 self.pending_speed = None; | support logic; impact depends on neighboring block and data flow.
| 550 |                 self.pending_dir = None; | support logic; impact depends on neighboring block and data flow. |
| 551 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 552 |             // Start filling new clutch | comment/documentation; changing affects understanding and intent trace
| 553 |             if self.clutch_state == ClutchState::Open { | runtime gate; condition changes can enable/disable cri
| 554 |                 self.clutch_state = ClutchState::Filling; | support logic; impact depends on neighboring block a
| 555 |                 self.clutch_phase_timer = 0.0; | support logic; impact depends on neighboring block and data flo
| 556 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 557 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 558 |         if self.shift_timer >= self.shift_duration { | runtime gate; condition changes can enable/disable critic
| 559 |             self.is_shifting = false; | support logic; impact depends on neighboring block and data flow. |
| 560 |             // Assess shift quality by clutch temperature rise | comment/documentation; changing affects unders
| 561 |             self.last_shift_qual = if self.clutch_temp_c > 150.0 { | support logic; impact depends on neighborin
| 562 |                 ShiftQuality::Harsh | support logic; impact depends on neighboring block and data flow. |
| 563 |             } else if self.clutch_temp_c > 100.0 { | support logic; impact depends on neighboring block and data
| 564 |                 ShiftQuality::Normal | support logic; impact depends on neighboring block and data flow. |
| 565 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 566 |                 ShiftQuality::Smooth | support logic; impact depends on neighboring block and data flow. |
| 567 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 568 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 569 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 570 | (blank) | blank separator; no runtime behavior. |
| 571 |     // ═══════════════════════════════════════════════════════════════════════════════ | comment/d
| 572 |     fn update_ivt(&mut self, _engine_rpm: f64, throttle_pct: f64, dt: f64) { | function signature/body; changes
| 573 |         // IVT: target ratio → target_speed = throttle_pct / 100 * MAX_SPEED | comment/documentation; changing
| 574 |         let target_spd = throttle_pct / 100.0 * MAX_SPEED_KMH; | support logic; impact depends on neighboring b
| 575 |         let current_spd = self.ground_speed_kmh; | support logic; impact depends on neighboring block and data f
| 576 |         let spd_err = target_spd - current_spd; | support logic; impact depends on neighboring block and data fl
| 577 |         // Vary IVT ratio to close speed error | comment/documentation; changing affects understanding and inter
| 578 |         let ratio_change = (spd_err * 0.01).clamp(-0.05, 0.05); | support logic; impact depends on neighboring b
| 579 |         self.ivt_ratio = (self.ivt_ratio + ratio_change * dt).clamp(0.0, 1.0); | support logic; impact depends o
| 580 |         // Effective gear ratio: 0 ratio → infinite, 1 ratio → direct | comment/documentation; changing affect
| 581 |         if self.ivt_ratio > 0.01 { | runtime gate; condition changes can enable/disable critical paths. |
| 582 |             self.gear_ratio = 50.0 / (self.ivt_ratio * 50.0); | support logic; impact depends on neighboring blo
| 583 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 584 |             self.gear_ratio = 999.0; | support logic; impact depends on neighboring block and data flow. |
| 585 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 586 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 587 | (blank) | blank separator; no runtime behavior. |
| 588 |     // ═══════════════════════════════════════════════════════════════════════════════ | comment/d
| 589 |     // Manual gear controls (called from main.rs keyboard input) | comment/documentation; changing affects under
| 590 |     pub fn manual_upshift(&mut self) { | function signature/body; changes affect API behavior and control-flow.
| 591 |         if self.auto_mode != AutoShiftMode::Auto { | runtime gate; condition changes can enable/disable critica
| 592 |             self.request_upshift(); | support logic; impact depends on neighboring block and data flow. |
| 593 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 594 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 595 |     pub fn manual_downshift(&mut self) { | function signature/body; changes affect API behavior and control-flo
| 596 |         if self.auto_mode != AutoShiftMode::Auto { | runtime gate; condition changes can enable/disable critica
| 597 |             self.request_downshift(); | support logic; impact depends on neighboring block and data flow. |
| 598 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 599 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 600 |     pub fn set_direction(&mut self, dir: Direction) { | function signature/body; changes affect API behavior an
| 601 |         self.begin_shift(self.range, self.speed_step, dir); | support logic; impact depends on neighboring bloc
| 602 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 603 |     pub fn set_neutral(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 604 |         self.direction = Direction::Neutral; | support logic; impact depends on neighboring block and data flow
| 605 |         self.is_neutral = true; | support logic; impact depends on neighboring block and data flow. |
| 606 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 607 |     pub fn toggle_creeper(&mut self) { | function signature/body; changes affect API behavior and control-flow.
| 608 |         self.creeper_engaged = !self.creeper_engaged; | support logic; impact depends on neighboring block and
| 609 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 610 |     pub fn toggle_auto(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 611 |         self.auto_mode = if self.auto_mode == AutoShiftMode::Auto { | support logic; impact depends on neighbor
| 612 |             AutoShiftMode::Manual | support logic; impact depends on neighboring block and data flow. |
| 613 |         } else { | support logic; impact depends on neighboring block and data flow. |
```

```
| 614 |             AutoShiftMode::Auto | support logic; impact depends on neighboring block and data flow. |
| 615 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 616 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 617 | (blank) | blank separator; no runtime behavior. |
| 618 |     fn compute_gear_label(&self) -> String { | function signature/body; changes affect API behavior and control-
| 619 |         match self.direction { | branch dispatcher; arm changes alter behavior class handling. |
| 620 |             Direction::Neutral => "N".into(), | support logic; impact depends on neighboring block and data flow
| 621 |             Direction::Park => "P".into(), | support logic; impact depends on neighboring block and data flow.
| 622 |             Direction::Forward => format!("{}{}", self.range, self.speed_step), | support logic; impact depends
| 623 |             Direction::Reverse => format!("R-{}{}", self.range, self.speed_step), | support logic; impact depend
| 624 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 625 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 626 | (blank) | blank separator; no runtime behavior. |
| 627 |     // ■ J1939 frame builders ■■■
```

```
| 690 |             let _ = tcm.tick(1340.0, 650.0, 30.0, 0.0, 0.1); | support logic; impact depends on neighboring bloc
| 691 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 692 | (blank) | blank separator; no runtime behavior. |
| 693 |         assert!(tcm.ground_speed_kmh > 20.0, "speed should build under partial throttle"); | support logic; impa
| 694 |         assert!( | support logic; impact depends on neighboring block and data flow. |
| 695 |             tcm.ground_speed_kmh < 40.0, | support logic; impact depends on neighboring block and data flow. |
| 696 |             "partial throttle should not saturate speedometer at 50 km/h" | support logic; impact depends on nei
| 697 |         ); | support logic; impact depends on neighboring block and data flow. |
| 698 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 699 | (blank) | blank separator; no runtime behavior. |
| 700 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 701 |     fn d4_full_throttle_reaches_transport_band_without_exceeding_limit() { | function signature/body; changes a
| 702 |         let mut tcm = EcuTcm::new(); | support logic; impact depends on neighboring block and data flow. |
| 703 |         tcm.direction = Direction::Forward; | support logic; impact depends on neighboring block and data flow.
| 704 |         tcm.range = GearRange::D; | support logic; impact depends on neighboring block and data flow. |
| 705 |         tcm.speed_step = 4; | support logic; impact depends on neighboring block and data flow. |
| 706 |         tcm.auto_mode = AutoShiftMode::Hold; | support logic; impact depends on neighboring block and data flow
| 707 |         tcm.clutch_state = ClutchState::Locked; | support logic; impact depends on neighboring block and data f
| 708 | (blank) | blank separator; no runtime behavior. |
| 709 |         for _ in 0..600 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 710 |             let _ = tcm.tick(2200.0, 1050.0, 100.0, 0.0, 0.1); | support logic; impact depends on neighboring b
| 711 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 712 | (blank) | blank separator; no runtime behavior. |
| 713 |         assert!(tcm.ground_speed_kmh >= 45.0, "transport gear should reach high-road band"); | support logic; i
| 714 |         assert!(tcm.ground_speed_kmh <= 50.0, "speed must remain capped at configured maximum"); | support logi
| 715 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 716 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/ecu_vcm.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 422 | Non-empty: 381 | Symbol count: 35

## Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| new | 12 |
| default | 4 |
| push | 3 |
| from_raw | 3 |
| build_id | 3 |
| compute_final | 2 |
| check_timeouts | 2 |
| update_from_frame | 2 |
| build_vcm_status | 2 |
| build_ws_master | 2 |
| build_dm13 | 2 |
| fmt | 1 |
| tick | 1 |

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 14 | const | VCM_SA | namespace and compile-time linkage |
| 18 | enum | VehicleMode | data/schema coupling and match/serde break risk |
| 28 | impl | std::fmt::Display for VehicleMode | method behavior and trait semantics |
| 29 | fn | fmt | API and behavior coupling to callers/implementers |
| 44 | struct | PowerDistribution | data/schema coupling and match/serde break risk |
| 56 | impl | Default for PowerDistribution | method behavior and trait semantics |
| 57 | fn | default | API and behavior coupling to callers/implementers |
| 62 | impl | PowerDistribution | method behavior and trait semantics |
| 63 | fn | new | API and behavior coupling to callers/implementers |
| 80 | struct | TorqueCoordination | data/schema coupling and match/serde break risk |
| 95 | impl | Default for TorqueCoordination | method behavior and trait semantics |
| 96 | fn | default | API and behavior coupling to callers/implementers |
| 101 | impl | TorqueCoordination | method behavior and trait semantics |
| 102 | fn | new | API and behavior coupling to callers/implementers |
| 113 | fn | compute_final | API and behavior coupling to callers/implementers |
| 131 | struct | CommHealth | data/schema coupling and match/serde break risk |
| 147 | impl | Default for CommHealth | method behavior and trait semantics |
| 148 | fn | default | API and behavior coupling to callers/implementers |

```
| 153 | impl | CommHealth | method behavior and trait semantics |
| 154 | fn | new | API and behavior coupling to callers/implementers |
| 173 | fn | check_timeouts | API and behavior coupling to callers/implementers |
| 174 | const | ECM_TIMEOUT | namespace and compile-time linkage |
| 175 | const | TCM_TIMEOUT | namespace and compile-time linkage |
| 176 | const | ABS_TIMEOUT | namespace and compile-time linkage |
| 177 | const | HCM_TIMEOUT | namespace and compile-time linkage |
| 200 | fn | update_from_frame | API and behavior coupling to callers/implementers |
| 213 | struct | EcuVcm | data/schema coupling and match/serde break risk |
| 241 | impl | Default for EcuVcm | method behavior and trait semantics |
| 242 | fn | default | API and behavior coupling to callers/implementers |
| 247 | impl | EcuVcm | method behavior and trait semantics |
| 248 | fn | new | API and behavior coupling to callers/implementers |
| 271 | fn | tick | API and behavior coupling to callers/implementers |
| 380 | fn | build_vcm_status | API and behavior coupling to callers/implementers |
| 401 | fn | build_ws_master | API and behavior coupling to callers/implementers |
| 411 | fn | build_dm13 | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Vehicle Control Module (VCM) / Gateway ECU — SA 0x14 (headway controller position). | comment/documentation; c
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! The VCM is the central coordinator in modern heavy machinery: | comment/documentation; changing affects unders
| 4 | //!   • Acts as CAN gateway between HS-CAN powertrain, MS-CAN body, and ISOBUS | comment/documentation; changing a
| 5 | //!   • Coordinates ADAS features across domains | comment/documentation; changing affects understanding and inter
| 6 | //!   • Implements vehicle-level safety functions (ISO 26262 ASIL-B) | comment/documentation; changing affects und
| 7 | //!   • Manages power distribution and load shedding | comment/documentation; changing affects understanding and i
| 8 | //!   • Provides torque coordination between ECM and braking systems | comment/documentation; changing affects und
| 9 | //!   • Hosts the proprietary J1939 working set master for ISOBUS | comment/documentation; changing affects unders
| 10 | //!   • Manages creep/crawl protection, rollaway prevention | comment/documentation; changing affects understandi
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | use crate::j1939::{addr, pgn, J1939Frame}; | import wiring; changing path/name can break compilation and module
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 | pub const VCM_SA: u8 = addr::HEADWAY; // 0x20 | support logic; impact depends on neighboring block and data flow
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | // ■■ Vehicle operating mode ■■■
```

```
| 55 | (blank) | blank separator; no runtime behavior. |
| 56 | impl Default for PowerDistribution { | implementation binding; behavior semantics and trait conformance live here
| 57 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 58 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 59 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 61 | (blank) | blank separator; no runtime behavior. |
| 62 | impl PowerDistribution { | implementation binding; behavior semantics and trait conformance live here. |
| 63 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 64 |         PowerDistribution { | support logic; impact depends on neighboring block and data flow. |
| 65 |             total_load_kw: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 66 |             engine_available_kw: 220.0, | support logic; impact depends on neighboring block and data flow. |
| 67 |             drivetrain_demand_kw: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 68 |             hydraulics_demand_kw: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 69 |             pto_demand_kw: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 70 |             electrical_demand_kw: 3.0, | support logic; impact depends on neighboring block and data flow. |
| 71 |             overload: false, | support logic; impact depends on neighboring block and data flow. |
| 72 |             load_shed_active: false, | support logic; impact depends on neighboring block and data flow. |
| 73 |             load_shed_reason: None, | support logic; impact depends on neighboring block and data flow. |
| 74 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 75 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 76 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 77 | (blank) | blank separator; no runtime behavior. |
| 78 | // ■■ Torque coordination ■■■
```

```
| 131 | pub struct CommHealth { | data layout contract; field changes can break callers/serde/tests. |
| 132 |     pub ecm_alive: bool, | support logic; impact depends on neighboring block and data flow. |
| 133 |     pub tcm_alive: bool, | support logic; impact depends on neighboring block and data flow. |
| 134 |     pub abs_alive: bool, | support logic; impact depends on neighboring block and data flow. |
| 135 |     pub bcm_alive: bool, | support logic; impact depends on neighboring block and data flow. |
| 136 |     pub icm_alive: bool, | support logic; impact depends on neighboring block and data flow. |
| 137 |     pub hcm_alive: bool, | support logic; impact depends on neighboring block and data flow. |
| 138 |     pub ecm_timeout_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 139 |     pub tcm_timeout_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 140 |     pub abs_timeout_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 141 |     pub last_ecm_ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 142 |     pub last_tcm_ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 143 |     pub last_abs_ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 144 |     pub last_hcm_ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 145 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 146 | (blank) | blank separator; no runtime behavior. |
| 147 | impl Default for CommHealth { | implementation binding; behavior semantics and trait conformance live here. |
| 148 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 149 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 150 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 151 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 152 | (blank) | blank separator; no runtime behavior. |
| 153 | impl CommHealth { | implementation binding; behavior semantics and trait conformance live here. |
| 154 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 155 |         CommHealth { | support logic; impact depends on neighboring block and data flow. |
| 156 |             ecm_alive: false, | support logic; impact depends on neighboring block and data flow. |
| 157 |             tcm_alive: false, | support logic; impact depends on neighboring block and data flow. |
| 158 |             abs_alive: false, | support logic; impact depends on neighboring block and data flow. |
| 159 |             bcm_alive: false, | support logic; impact depends on neighboring block and data flow. |
| 160 |             icm_alive: false, | support logic; impact depends on neighboring block and data flow. |
| 161 |             hcm_alive: false, | support logic; impact depends on neighboring block and data flow. |
| 162 |             ecm_timeout_count: 0, | support logic; impact depends on neighboring block and data flow. |
| 163 |             tcm_timeout_count: 0, | support logic; impact depends on neighboring block and data flow. |
| 164 |             abs_timeout_count: 0, | support logic; impact depends on neighboring block and data flow. |
| 165 |             last_ecm_ts: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 166 |             last_tcm_ts: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 167 |             last_abs_ts: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 168 |             last_hcm_ts: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 169 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 170 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 171 | (blank) | blank separator; no runtime behavior. |
| 172 |     /// Check for message timeouts — real VCM monitors each critical ECU | comment/documentation; changing affec
| 173 |     pub fn check_timeouts(&mut self, elapsed: f64) { | function signature/body; changes affect API behavior and
| 174 |         const ECM_TIMEOUT: f64 = 0.050; // EEC1 at 10ms → 50ms timeout | support logic; impact depends on neigh
| 175 |         const TCM_TIMEOUT: f64 = 0.100; // ETC1 at 20ms → 100ms timeout | support logic; impact depends on neig
| 176 |         const ABS_TIMEOUT: f64 = 0.200; // EBC1 at 20ms → 200ms timeout | support logic; impact depends on neig
| 177 |         const HCM_TIMEOUT: f64 = 0.500; // HITCH at 100ms → 500ms timeout | support logic; impact depends on ne
| 178 | (blank) | blank separator; no runtime behavior. |
| 179 |         let was_ecm = self.ecm_alive; | support logic; impact depends on neighboring block and data flow. |
| 180 |         self.ecm_alive = (elapsed - self.last_ecm_ts) < ECM_TIMEOUT; | support logic; impact depends on neighbo
| 181 |         if was_ecm && !self.ecm_alive { | runtime gate; condition changes can enable/disable critical paths. |
| 182 |             self.ecm_timeout_count += 1; | support logic; impact depends on neighboring block and data flow. |
| 183 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 184 | (blank) | blank separator; no runtime behavior. |
| 185 |         let was_tcm = self.tcm_alive; | support logic; impact depends on neighboring block and data flow. |
| 186 |         self.tcm_alive = (elapsed - self.last_tcm_ts) < TCM_TIMEOUT; | support logic; impact depends on neighbo
| 187 |         if was_tcm && !self.tcm_alive { | runtime gate; condition changes can enable/disable critical paths. |
| 188 |             self.tcm_timeout_count += 1; | support logic; impact depends on neighboring block and data flow. |
| 189 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 190 | (blank) | blank separator; no runtime behavior. |
| 191 |         let was_abs = self.abs_alive; | support logic; impact depends on neighboring block and data flow. |
| 192 |         self.abs_alive = (elapsed - self.last_abs_ts) < ABS_TIMEOUT; | support logic; impact depends on neighbo
| 193 |         if was_abs && !self.abs_alive { | runtime gate; condition changes can enable/disable critical paths. |
| 194 |             self.abs_timeout_count += 1; | support logic; impact depends on neighboring block and data flow. |
| 195 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 196 | (blank) | blank separator; no runtime behavior. |
| 197 |         self.hcm_alive = (elapsed - self.last_hcm_ts) < HCM_TIMEOUT; | support logic; impact depends on neighbo
| 198 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 199 | (blank) | blank separator; no runtime behavior. |
| 200 |     pub fn update_from_frame(&mut self, frame: &J1939Frame) { | function signature/body; changes affect API beha
| 201 |         let ts = frame.timestamp; | support logic; impact depends on neighboring block and data flow. |
| 202 |         match frame.sa { | branch dispatcher; arm changes alter behavior class handling. |
| 203 |             s if s == addr::ECM_1 => self.last_ecm_ts = ts, | support logic; impact depends on neighboring bloc
| 204 |             s if s == addr::TRANSMISSION => self.last_tcm_ts = ts, | support logic; impact depends on neighborin
| 205 |             s if s == addr::BRAKES => self.last_abs_ts = ts, | support logic; impact depends on neighboring bloc
| 206 |             s if s == addr::HITCH => self.last_hcm_ts = ts, | support logic; impact depends on neighboring bloc
```

```
| 207 |             _ => {} | support logic; impact depends on neighboring block and data flow. |
| 208 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 209 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 210 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 211 | (blank) | blank separator; no runtime behavior. |
| 212 | // ■■ VCM ■■■
```

```
| 283 |         received_frames: &[J1939Frame], | support logic; impact depends on neighboring block and data flow. |
| 284 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 285 |     ) -> Vec<J1939Frame> { | support logic; impact depends on neighboring block and data flow. |
| 286 |         // ■ Update comms health ■■■
```

```
| 359 |         self.t_dm13 += dt; | support logic; impact depends on neighboring block and data flow. |
| 360 |         let mut frames: Vec<J1939Frame> = Vec::new(); | support logic; impact depends on neighboring block and d
| 361 | (blank) | blank separator; no runtime behavior. |
| 362 |         if self.t_status >= 0.100 { | runtime gate; condition changes can enable/disable critical paths. |
| 363 |             self.t_status = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 364 |             frames.push(self.build_vcm_status(elapsed)); | support logic; impact depends on neighboring block an
| 365 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 366 |         if self.t_ws_mst >= 0.100 { | runtime gate; condition changes can enable/disable critical paths. |
| 367 |             self.t_ws_mst = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 368 |             frames.push(self.build_ws_master(elapsed)); | support logic; impact depends on neighboring block and
| 369 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 370 |         if self.t_dm13 >= 5.000 { | runtime gate; condition changes can enable/disable critical paths. |
| 371 |             self.t_dm13 = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 372 |             frames.push(self.build_dm13(elapsed)); | support logic; impact depends on neighboring block and data
| 373 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 374 |         frames | support logic; impact depends on neighboring block and data flow. |
| 375 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 376 | (blank) | blank separator; no runtime behavior. |
| 377 |     // ■ Frame builders ■■■
```

```
|---|---:|
| new | 17 |
| default | 6 |
| tick | 4 |
| powered_sas | 4 |
| fmt | 3 |
| capacitance_nf_per_m | 2 |
| wire_resistance_ohm | 2 |
| requested_current_for_branch | 2 |
| set_branch_fault | 2 |
| set_control_fault | 2 |
| branch | 2 |
| reset_faults | 1 |
| ignition_relay_controls_keyed_ecu_power | 1 |
| short_to_ground_blows_ecu_branch | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 4 | enum | ElectricalLine | data/schema coupling and match/serde break risk |
| 13 | impl | std::fmt::Display for ElectricalLine | method behavior and trait semantics |
| 14 | fn | fmt | API and behavior coupling to callers/implementers |
| 28 | enum | ElectricalBranchKind | data/schema coupling and match/serde break risk |
| 35 | enum | ElectricalFaultMode | data/schema coupling and match/serde break risk |
| 42 | impl | std::fmt::Display for ElectricalFaultMode | method behavior and trait semantics |
| 43 | fn | fmt | API and behavior coupling to callers/implementers |
| 55 | enum | ElectricalControlPoint | data/schema coupling and match/serde break risk |
| 61 | impl | std::fmt::Display for ElectricalControlPoint | method behavior and trait semantics |
| 62 | fn | fmt | API and behavior coupling to callers/implementers |
| 73 | struct | ElectricalBranch | data/schema coupling and match/serde break risk |
| 95 | impl | ElectricalBranch | method behavior and trait semantics |
| 97 | fn | new | API and behavior coupling to callers/implementers |
| 137 | struct | ElectricalRelayState | data/schema coupling and match/serde break risk |
| 143 | struct | ElectricalSnapshot | data/schema coupling and match/serde break risk |
| 153 | impl | Default for ElectricalSnapshot | method behavior and trait semantics |
| 154 | fn | default | API and behavior coupling to callers/implementers |
| 168 | struct | ElectricalLoadRequest | data/schema coupling and match/serde break risk |
| 182 | struct | ElectricalSystem | data/schema coupling and match/serde break risk |
| 191 | impl | Default for ElectricalSystem | method behavior and trait semantics |
| 192 | fn | default | API and behavior coupling to callers/implementers |
| 197 | impl | ElectricalSystem | method behavior and trait semantics |
| 198 | fn | new | API and behavior coupling to callers/implementers |
| 222 | fn | wire_resistance_ohm | API and behavior coupling to callers/implementers |
| 228 | fn | capacitance_nf_per_m | API and behavior coupling to callers/implementers |
| 242 | fn | requested_current_for_branch | API and behavior coupling to callers/implementers |
| 267 | fn | set_branch_fault | API and behavior coupling to callers/implementers |
| 276 | fn | set_control_fault | API and behavior coupling to callers/implementers |
| 284 | fn | reset_faults | API and behavior coupling to callers/implementers |
| 294 | fn | tick | API and behavior coupling to callers/implementers |
| 400 | fn | powered_sas | API and behavior coupling to callers/implementers |
| 413 | fn | branch | API and behavior coupling to callers/implementers |
| 419 | module | tests | namespace and compile-time linkage |
| 424 | fn | ignition_relay_controls_keyed_ecu_power | API and behavior coupling to callers/implementers |
| 445 | fn | short_to_ground_blows_ecu_branch | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use crate::{ecu_bcm::Fuse, IgnitionState}; | import wiring; changing path/name can break compilation and module li
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent tracea
| 4 | pub enum ElectricalLine { | state machine variants; changes ripple through matches and business logic. |
| 5 |     Line30, | support logic; impact depends on neighboring block and data flow. |
| 6 |     Line15Acc, | support logic; impact depends on neighboring block and data flow. |
| 7 |     Line15Ign, | support logic; impact depends on neighboring block and data flow. |
| 8 |     Line31, | support logic; impact depends on neighboring block and data flow. |
| 9 |     HsCan, | support logic; impact depends on neighboring block and data flow. |
| 10 |     MsCan, | support logic; impact depends on neighboring block and data flow. |
| 11 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | impl std::fmt::Display for ElectricalLine { | implementation binding; behavior semantics and trait conformance li
| 14 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 15 |         let s = match self { | support logic; impact depends on neighboring block and data flow. |
```

```
| 16 |             ElectricalLine::Line30 => "30", | support logic; impact depends on neighboring block and data flow.
| 17 |             ElectricalLine::Line15Acc => "15-ACC", | support logic; impact depends on neighboring block and data
| 18 |             ElectricalLine::Line15Ign => "15-IGN", | support logic; impact depends on neighboring block and data
| 19 |             ElectricalLine::Line31 => "31-GND", | support logic; impact depends on neighboring block and data flo
| 20 |             ElectricalLine::HsCan => "HS-CAN", | support logic; impact depends on neighboring block and data flow
| 21 |             ElectricalLine::MsCan => "MS-CAN", | support logic; impact depends on neighboring block and data flow
| 22 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 24 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 25 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 28 | pub enum ElectricalBranchKind { | state machine variants; changes ripple through matches and business logic. |
| 29 |     Supply, | support logic; impact depends on neighboring block and data flow. |
| 30 |     Load, | support logic; impact depends on neighboring block and data flow. |
| 31 |     Network, | support logic; impact depends on neighboring block and data flow. |
| 32 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 35 | pub enum ElectricalFaultMode { | state machine variants; changes ripple through matches and business logic. |
| 36 |     None, | support logic; impact depends on neighboring block and data flow. |
| 37 |     OpenCircuit, | support logic; impact depends on neighboring block and data flow. |
| 38 |     ShortToGround, | support logic; impact depends on neighboring block and data flow. |
| 39 |     HighResistance, | support logic; impact depends on neighboring block and data flow. |
| 40 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 41 | (blank) | blank separator; no runtime behavior. |
| 42 | impl std::fmt::Display for ElectricalFaultMode { | implementation binding; behavior semantics and trait conformar
| 43 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 44 |         let s = match self { | support logic; impact depends on neighboring block and data flow. |
| 45 |             ElectricalFaultMode::None => "OK", | support logic; impact depends on neighboring block and data flow
| 46 |             ElectricalFaultMode::OpenCircuit => "OPEN", | support logic; impact depends on neighboring block and
| 47 |             ElectricalFaultMode::ShortToGround => "SHORT-GND", | support logic; impact depends on neighboring blo
| 48 |             ElectricalFaultMode::HighResistance => "HI-R", | support logic; impact depends on neighboring block a
| 49 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 50 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 51 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 53 | (blank) | blank separator; no runtime behavior. |
| 54 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 55 | pub enum ElectricalControlPoint { | state machine variants; changes ripple through matches and business logic. |
| 56 |     MainGround, | support logic; impact depends on neighboring block and data flow. |
| 57 |     AccessoryRelay, | support logic; impact depends on neighboring block and data flow. |
| 58 |     IgnitionRelay, | support logic; impact depends on neighboring block and data flow. |
| 59 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 | impl std::fmt::Display for ElectricalControlPoint { | implementation binding; behavior semantics and trait confor
| 62 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 63 |         let s = match self { | support logic; impact depends on neighboring block and data flow. |
| 64 |             ElectricalControlPoint::MainGround => "MAIN-GND", | support logic; impact depends on neighboring bloc
| 65 |             ElectricalControlPoint::AccessoryRelay => "ACC-RELAY", | support logic; impact depends on neighboring
| 66 |             ElectricalControlPoint::IgnitionRelay => "IGN-RELAY", | support logic; impact depends on neighboring
| 67 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 68 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 69 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 70 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 | (blank) | blank separator; no runtime behavior. |
| 72 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 73 | pub struct ElectricalBranch { | data layout contract; field changes can break callers/serde/tests. |
| 74 |     pub id: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 75 |     pub from: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 76 |     pub to: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 77 |     pub line: ElectricalLine, | support logic; impact depends on neighboring block and data flow. |
| 78 |     pub kind: ElectricalBranchKind, | support logic; impact depends on neighboring block and data flow. |
| 79 |     pub fuse_id: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 80 |     pub fuse_rating_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 81 |     pub gauge_mm2: f64, | support logic; impact depends on neighboring block and data flow. |
| 82 |     pub length_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 83 |     pub requested_current_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 84 |     pub actual_current_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 85 |     pub voltage_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 86 |     pub capacitance_nf: f64, | support logic; impact depends on neighboring block and data flow. |
| 87 |     pub resistance_ohm: f64, | support logic; impact depends on neighboring block and data flow. |
| 88 |     pub fault: ElectricalFaultMode, | support logic; impact depends on neighboring block and data flow. |
| 89 |     pub blown: bool, | support logic; impact depends on neighboring block and data flow. |
| 90 |     pub energized: bool, | support logic; impact depends on neighboring block and data flow. |
| 91 |     pub target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
```

```
| 92 |     pub note: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 93 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 94 | (blank) | blank separator; no runtime behavior. |
| 95 | impl ElectricalBranch { | implementation binding; behavior semantics and trait conformance live here. |
| 96 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent trace
| 97 |     fn new( | function signature/body; changes affect API behavior and control-flow. |
| 98 |         id: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 99 |         from: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 100 |         to: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 101 |         line: ElectricalLine, | support logic; impact depends on neighboring block and data flow. |
| 102 |         kind: ElectricalBranchKind, | support logic; impact depends on neighboring block and data flow. |
| 103 |         fuse_id: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 104 |         fuse_rating_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 105 |         gauge_mm2: f64, | support logic; impact depends on neighboring block and data flow. |
| 106 |         length_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 107 |         target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 108 |         note: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 109 |     ) -> Self { | support logic; impact depends on neighboring block and data flow. |
| 110 |         let capacitance_nf = length_m * ElectricalSystem::capacitance_nf_per_m(gauge_mm2, kind == ElectricalBra
| 111 |         let resistance_ohm = ElectricalSystem::wire_resistance_ohm(length_m, gauge_mm2, kind == ElectricalBranc
| 112 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 113 |             id, | support logic; impact depends on neighboring block and data flow. |
| 114 |             from, | support logic; impact depends on neighboring block and data flow. |
| 115 |             to, | support logic; impact depends on neighboring block and data flow. |
| 116 |             line, | support logic; impact depends on neighboring block and data flow. |
| 117 |             kind, | support logic; impact depends on neighboring block and data flow. |
| 118 |             fuse_id, | support logic; impact depends on neighboring block and data flow. |
| 119 |             fuse_rating_a, | support logic; impact depends on neighboring block and data flow. |
| 120 |             gauge_mm2, | support logic; impact depends on neighboring block and data flow. |
| 121 |             length_m, | support logic; impact depends on neighboring block and data flow. |
| 122 |             requested_current_a: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 123 |             actual_current_a: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 124 |             voltage_v: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 125 |             capacitance_nf, | support logic; impact depends on neighboring block and data flow. |
| 126 |             resistance_ohm, | support logic; impact depends on neighboring block and data flow. |
| 127 |             fault: ElectricalFaultMode::None, | support logic; impact depends on neighboring block and data flo
| 128 |             blown: false, | support logic; impact depends on neighboring block and data flow. |
| 129 |             energized: false, | support logic; impact depends on neighboring block and data flow. |
| 130 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 131 |             note, | support logic; impact depends on neighboring block and data flow. |
| 132 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 133 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 134 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 135 | (blank) | blank separator; no runtime behavior. |
| 136 | #[derive(Debug, Clone, Default)] | comment/documentation; changing affects understanding and intent traceabilit
| 137 | pub struct ElectricalRelayState { | data layout contract; field changes can break callers/serde/tests. |
| 138 |     pub accessory_closed: bool, | support logic; impact depends on neighboring block and data flow. |
| 139 |     pub ignition_closed: bool, | support logic; impact depends on neighboring block and data flow. |
| 140 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 141 | (blank) | blank separator; no runtime behavior. |
| 142 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 143 | pub struct ElectricalSnapshot { | data layout contract; field changes can break callers/serde/tests. |
| 144 |     pub line30_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 145 |     pub line15_acc_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 146 |     pub line15_ign_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 147 |     pub line31_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 148 |     pub source_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 149 |     pub total_current_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 150 |     pub ground_drop_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 151 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 152 | (blank) | blank separator; no runtime behavior. |
| 153 | impl Default for ElectricalSnapshot { | implementation binding; behavior semantics and trait conformance live h
| 154 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 155 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 156 |             line30_v: 12.6, | support logic; impact depends on neighboring block and data flow. |
| 157 |             line15_acc_v: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 158 |             line15_ign_v: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 159 |             line31_v: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 160 |             source_v: 12.6, | support logic; impact depends on neighboring block and data flow. |
| 161 |             total_current_a: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 162 |             ground_drop_v: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 163 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 164 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 165 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 | (blank) | blank separator; no runtime behavior. |
| 167 | #[derive(Debug, Clone, Default)] | comment/documentation; changing affects understanding and intent traceabilit
```

```
| 168 | pub struct ElectricalLoadRequest { | data layout contract; field changes can break callers/serde/tests. |
| 169 |     pub bcm_supply_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 170 |     pub vcm_supply_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 171 |     pub ecm_supply_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 172 |     pub tcm_supply_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 173 |     pub abs_supply_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 174 |     pub hcm_supply_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 175 |     pub icm_supply_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 176 |     pub body_load_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 177 |     pub hs_can_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 178 |     pub ms_can_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 179 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 180 | (blank) | blank separator; no runtime behavior. |
| 181 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 182 | pub struct ElectricalSystem { | data layout contract; field changes can break callers/serde/tests. |
| 183 |     pub relays: ElectricalRelayState, | support logic; impact depends on neighboring block and data flow. |
| 184 |     pub snapshot: ElectricalSnapshot, | support logic; impact depends on neighboring block and data flow. |
| 185 |     pub branches: Vec<ElectricalBranch>, | support logic; impact depends on neighboring block and data flow. |
| 186 |     pub main_ground_fault: ElectricalFaultMode, | support logic; impact depends on neighboring block and data f
| 187 |     pub accessory_relay_fault: ElectricalFaultMode, | support logic; impact depends on neighboring block and da
| 188 |     pub ignition_relay_fault: ElectricalFaultMode, | support logic; impact depends on neighboring block and data
| 189 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 190 | (blank) | blank separator; no runtime behavior. |
| 191 | impl Default for ElectricalSystem { | implementation binding; behavior semantics and trait conformance live here
| 192 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 193 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 194 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 195 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 196 | (blank) | blank separator; no runtime behavior. |
| 197 | impl ElectricalSystem { | implementation binding; behavior semantics and trait conformance live here. |
| 198 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 199 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 200 |             relays: ElectricalRelayState::default(), | support logic; impact depends on neighboring block and da
| 201 |             snapshot: ElectricalSnapshot::default(), | support logic; impact depends on neighboring block and da
| 202 |             branches: vec![ | support logic; impact depends on neighboring block and data flow. |
| 203 |                 ElectricalBranch::new("BAT-MAIN", "Battery", "Main Fuse Box", ElectricalLine::Line30, Electrical
| 204 |                 ElectricalBranch::new("ALT-B+", "Alternator", "Battery/Main Bus", ElectricalLine::Line30, Electr
| 205 |                 ElectricalBranch::new("BCM-PWR", "Main Fuse Box", "BCM", ElectricalLine::Line15Acc, ElectricalBr
| 206 |                 ElectricalBranch::new("ICM-PWR", "BCM", "ICM", ElectricalLine::Line15Acc, ElectricalBranchKind:
| 207 |                 ElectricalBranch::new("VCM-PWR", "Main Fuse Box", "VCM", ElectricalLine::Line15Ign, ElectricalBr
| 208 |                 ElectricalBranch::new("ECM-PWR", "Main Fuse Box", "ECM", ElectricalLine::Line15Ign, ElectricalBr
| 209 |                 ElectricalBranch::new("TCM-PWR", "Main Fuse Box", "TCM", ElectricalLine::Line15Ign, ElectricalBr
| 210 |                 ElectricalBranch::new("ABS-PWR", "Main Fuse Box", "ABS/ESP", ElectricalLine::Line15Ign, Electric
| 211 |                 ElectricalBranch::new("HCM-PWR", "Main Fuse Box", "HCM", ElectricalLine::Line15Ign, ElectricalBr
| 212 |                 ElectricalBranch::new("BODY-LOAD", "BCM", "Lights/HVAC/Wiper/Horn", ElectricalLine::Line15Acc, E
| 213 |                 ElectricalBranch::new("HS-CAN-1", "VCM", "ECM/TCM/ABS/HCM", ElectricalLine::HsCan, ElectricalBra
| 214 |                 ElectricalBranch::new("MS-CAN-BODY", "BCM", "ICM/body domain", ElectricalLine::MsCan, Electrical
| 215 |             ], | support logic; impact depends on neighboring block and data flow. |
| 216 |             main_ground_fault: ElectricalFaultMode::None, | support logic; impact depends on neighboring block a
| 217 |             accessory_relay_fault: ElectricalFaultMode::None, | support logic; impact depends on neighboring bl
| 218 |             ignition_relay_fault: ElectricalFaultMode::None, | support logic; impact depends on neighboring bloc
| 219 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 220 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 221 | (blank) | blank separator; no runtime behavior. |
| 222 |     pub fn wire_resistance_ohm(length_m: f64, gauge_mm2: f64, twisted_pair: bool) -> f64 { | function signature/
| 223 |         let rho = 0.0175; | support logic; impact depends on neighboring block and data flow. |
| 224 |         let loop_factor = if twisted_pair { 2.0 } else { 2.2 }; | support logic; impact depends on neighboring b
| 225 |         (rho * length_m * loop_factor / gauge_mm2.max(0.35)).max(0.0005) | support logic; impact depends on neig
| 226 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 227 | (blank) | blank separator; no runtime behavior. |
| 228 |     pub fn capacitance_nf_per_m(gauge_mm2: f64, twisted_pair: bool) -> f64 { | function signature/body; changes
| 229 |         if twisted_pair { | runtime gate; condition changes can enable/disable critical paths. |
| 230 |             0.05 | support logic; impact depends on neighboring block and data flow. |
| 231 |         } else if gauge_mm2 <= 1.0 { | support logic; impact depends on neighboring block and data flow. |
| 232 |             0.09 | support logic; impact depends on neighboring block and data flow. |
| 233 |         } else if gauge_mm2 <= 2.5 { | support logic; impact depends on neighboring block and data flow. |
| 234 |             0.11 | support logic; impact depends on neighboring block and data flow. |
| 235 |         } else if gauge_mm2 <= 6.0 { | support logic; impact depends on neighboring block and data flow. |
| 236 |             0.13 | support logic; impact depends on neighboring block and data flow. |
| 237 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 238 |             0.16 | support logic; impact depends on neighboring block and data flow. |
| 239 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 240 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 241 | (blank) | blank separator; no runtime behavior. |
| 242 |     fn requested_current_for_branch(branch_id: &str, loads: &ElectricalLoadRequest) -> f64 { | function signatu
| 243 |         match branch_id { | branch dispatcher; arm changes alter behavior class handling. |
```

```
| 244 |             "BAT-MAIN" => loads.bcm_supply_a | support logic; impact depends on neighboring block and data flow
| 245 |                 + loads.vcm_supply_a | support logic; impact depends on neighboring block and data flow. |
| 246 |                 + loads.ecm_supply_a | support logic; impact depends on neighboring block and data flow. |
| 247 |                 + loads.tcm_supply_a | support logic; impact depends on neighboring block and data flow. |
| 248 |                 + loads.abs_supply_a | support logic; impact depends on neighboring block and data flow. |
| 249 |                 + loads.hcm_supply_a | support logic; impact depends on neighboring block and data flow. |
| 250 |                 + loads.icm_supply_a | support logic; impact depends on neighboring block and data flow. |
| 251 |                 + loads.body_load_a, | support logic; impact depends on neighboring block and data flow. |
| 252 |             "ALT-B+" => 0.0, | support logic; impact depends on neighboring block and data flow. |
| 253 |             "BCM-PWR" => loads.bcm_supply_a, | support logic; impact depends on neighboring block and data flow
| 254 |             "ICM-PWR" => loads.icm_supply_a, | support logic; impact depends on neighboring block and data flow
| 255 |             "VCM-PWR" => loads.vcm_supply_a, | support logic; impact depends on neighboring block and data flow
| 256 |             "ECM-PWR" => loads.ecm_supply_a, | support logic; impact depends on neighboring block and data flow
| 257 |             "TCM-PWR" => loads.tcm_supply_a, | support logic; impact depends on neighboring block and data flow
| 258 |             "ABS-PWR" => loads.abs_supply_a, | support logic; impact depends on neighboring block and data flow
| 259 |             "HCM-PWR" => loads.hcm_supply_a, | support logic; impact depends on neighboring block and data flow
| 260 |             "BODY-LOAD" => loads.body_load_a, | support logic; impact depends on neighboring block and data flo
| 261 |             "HS-CAN-1" => loads.hs_can_a, | support logic; impact depends on neighboring block and data flow. |
| 262 |             "MS-CAN-BODY" => loads.ms_can_a, | support logic; impact depends on neighboring block and data flow
| 263 |             _ => 0.0, | support logic; impact depends on neighboring block and data flow. |
| 264 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 265 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 266 | (blank) | blank separator; no runtime behavior. |
| 267 |     pub fn set_branch_fault(&mut self, branch_id: &str, fault: ElectricalFaultMode) { | function signature/body
| 268 |         if let Some(branch) = self.branches.iter_mut().find(\|b\| b.id == branch_id) { | runtime gate; conditio
| 269 |             branch.fault = fault; | support logic; impact depends on neighboring block and data flow. |
| 270 |             if fault == ElectricalFaultMode::None { | runtime gate; condition changes can enable/disable critica
| 271 |                 branch.blown = false; | support logic; impact depends on neighboring block and data flow. |
| 272 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 273 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 274 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 275 | (blank) | blank separator; no runtime behavior. |
| 276 |     pub fn set_control_fault(&mut self, point: ElectricalControlPoint, fault: ElectricalFaultMode) { | function
| 277 |         match point { | branch dispatcher; arm changes alter behavior class handling. |
| 278 |             ElectricalControlPoint::MainGround => self.main_ground_fault = fault, | support logic; impact depen
| 279 |             ElectricalControlPoint::AccessoryRelay => self.accessory_relay_fault = fault, | support logic; impac
| 280 |             ElectricalControlPoint::IgnitionRelay => self.ignition_relay_fault = fault, | support logic; impact
| 281 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 282 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 283 | (blank) | blank separator; no runtime behavior. |
| 284 |     pub fn reset_faults(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 285 |         self.main_ground_fault = ElectricalFaultMode::None; | support logic; impact depends on neighboring bloc
| 286 |         self.accessory_relay_fault = ElectricalFaultMode::None; | support logic; impact depends on neighboring b
| 287 |         self.ignition_relay_fault = ElectricalFaultMode::None; | support logic; impact depends on neighboring b
| 288 |         for branch in &mut self.branches { | iteration logic; may alter timing, throughput, and safety constrai
| 289 |             branch.fault = ElectricalFaultMode::None; | support logic; impact depends on neighboring block and
| 290 |             branch.blown = false; | support logic; impact depends on neighboring block and data flow. |
| 291 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 292 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 293 | (blank) | blank separator; no runtime behavior. |
| 294 |     pub fn tick( | function signature/body; changes affect API behavior and control-flow. |
| 295 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 296 |         ignition: IgnitionState, | support logic; impact depends on neighboring block and data flow. |
| 297 |         source_voltage_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 298 |         alternator_voltage_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 299 |         loads: &ElectricalLoadRequest, | support logic; impact depends on neighboring block and data flow. |
| 300 |         bcm_fuses: &[Fuse], | support logic; impact depends on neighboring block and data flow. |
| 301 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 302 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 303 |         self.relays.accessory_closed = matches!(ignition, IgnitionState::Accessory \| IgnitionState::On \| Igni
| 304 |             && self.accessory_relay_fault != ElectricalFaultMode::OpenCircuit; | support logic; impact depends
| 305 |         self.relays.ignition_closed = matches!(ignition, IgnitionState::On \| IgnitionState::Cranking \| Ignitio
| 306 |             && self.ignition_relay_fault != ElectricalFaultMode::OpenCircuit; | support logic; impact depends o
| 307 | (blank) | blank separator; no runtime behavior. |
| 308 |         let ground_r = match self.main_ground_fault { | support logic; impact depends on neighboring block and
| 309 |             ElectricalFaultMode::None => 0.004, | support logic; impact depends on neighboring block and data f
| 310 |             ElectricalFaultMode::HighResistance => 0.09, | support logic; impact depends on neighboring block a
| 311 |             ElectricalFaultMode::OpenCircuit => 1.5, | support logic; impact depends on neighboring block and d
| 312 |             ElectricalFaultMode::ShortToGround => 0.004, | support logic; impact depends on neighboring block a
| 313 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 314 | (blank) | blank separator; no runtime behavior. |
| 315 |         self.snapshot.source_v = source_voltage_v.max(alternator_voltage_v).max(0.0); | support logic; impact d
| 316 |         self.snapshot.line30_v = self.snapshot.source_v; | support logic; impact depends on neighboring block a
| 317 |         self.snapshot.line15_acc_v = if self.relays.accessory_closed { | support logic; impact depends on neigh
| 318 |             (self.snapshot.line30_v - if self.accessory_relay_fault == ElectricalFaultMode::HighResistance { 1.8
| 319 |         } else { | support logic; impact depends on neighboring block and data flow. |
```

```
| 320 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 321 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 322 |         self.snapshot.line15_ign_v = if self.relays.ignition_closed { | support logic; impact depends on neighbo
| 323 |             (self.snapshot.line30_v - if self.ignition_relay_fault == ElectricalFaultMode::HighResistance { 2.2
| 324 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 325 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 326 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 327 | (blank) | blank separator; no runtime behavior. |
| 328 |         let line30_v = self.snapshot.line30_v; | support logic; impact depends on neighboring block and data flo
| 329 |         let line15_acc_v = self.snapshot.line15_acc_v; | support logic; impact depends on neighboring block and
| 330 |         let line15_ign_v = self.snapshot.line15_ign_v; | support logic; impact depends on neighboring block and
| 331 |         let line31_v = self.snapshot.line31_v; | support logic; impact depends on neighboring block and data flo
| 332 |         let mut total_current = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 333 |         for branch in &mut self.branches { | iteration logic; may alter timing, throughput, and safety constrain
| 334 |             branch.requested_current_a = Self::requested_current_for_branch(branch.id, loads); | support logic;
| 335 | (blank) | blank separator; no runtime behavior. |
| 336 |             if let Some(fuse) = bcm_fuses.iter().find(\|f\| f.id == branch.id \|\| f.id == branch.fuse_id) { | r
| 337 |                 if fuse.blown { | runtime gate; condition changes can enable/disable critical paths. |
| 338 |                     branch.blown = true; | support logic; impact depends on neighboring block and data flow. |
| 339 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 341 | (blank) | blank separator; no runtime behavior. |
| 342 |             let source_v = match branch.line { | support logic; impact depends on neighboring block and data flo
| 343 |                 ElectricalLine::Line30 => line30_v, | support logic; impact depends on neighboring block and dat
| 344 |                 ElectricalLine::Line15Acc => line15_acc_v, | support logic; impact depends on neighboring block
| 345 |                 ElectricalLine::Line15Ign => line15_ign_v, | support logic; impact depends on neighboring block
| 346 |                 ElectricalLine::Line31 => line31_v, | support logic; impact depends on neighboring block and dat
| 347 |                 ElectricalLine::HsCan => line15_ign_v, | support logic; impact depends on neighboring block and
| 348 |                 ElectricalLine::MsCan => line15_acc_v, | support logic; impact depends on neighboring block and
| 349 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 350 |             let extra_r = match branch.fault { | support logic; impact depends on neighboring block and data flo
| 351 |                 ElectricalFaultMode::None => 0.0, | support logic; impact depends on neighboring block and data
| 352 |                 ElectricalFaultMode::HighResistance => 0.35, | support logic; impact depends on neighboring bloc
| 353 |                 ElectricalFaultMode::OpenCircuit => 1.0e9, | support logic; impact depends on neighboring block
| 354 |                 ElectricalFaultMode::ShortToGround => 0.02, | support logic; impact depends on neighboring bloc
| 355 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 356 | (blank) | blank separator; no runtime behavior. |
| 357 |             if branch.fault == ElectricalFaultMode::OpenCircuit \|\| branch.blown \|\| source_v < 1.0 { | runtin
| 358 |                 branch.actual_current_a = 0.0; | support logic; impact depends on neighboring block and data flo
| 359 |                 branch.energized = false; | support logic; impact depends on neighboring block and data flow. |
| 360 |                 branch.voltage_v += (0.0 - branch.voltage_v) * (dt / 0.04).min(1.0); | support logic; impact dep
| 361 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 362 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 363 | (blank) | blank separator; no runtime behavior. |
| 364 |             if branch.fault == ElectricalFaultMode::ShortToGround { | runtime gate; condition changes can enable
| 365 |                 let short_i = source_v / (branch.resistance_ohm + ground_r + extra_r).max(0.005); | support logi
| 366 |                 if short_i > branch.fuse_rating_a * 1.15 { | runtime gate; condition changes can enable/disable
| 367 |                     branch.blown = true; | support logic; impact depends on neighboring block and data flow. |
| 368 |                     branch.actual_current_a = 0.0; | support logic; impact depends on neighboring block and data
| 369 |                     branch.energized = false; | support logic; impact depends on neighboring block and data flow
| 370 |                     branch.voltage_v = 0.0; | support logic; impact depends on neighboring block and data flow.
| 371 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 372 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 373 |                 branch.actual_current_a = short_i; | support logic; impact depends on neighboring block and data
| 374 |                 branch.energized = true; | support logic; impact depends on neighboring block and data flow. |
| 375 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 376 |                 branch.actual_current_a = branch.requested_current_a.max(0.0); | support logic; impact depends
| 377 |                 branch.energized = source_v > 9.0 \|\| branch.kind == ElectricalBranchKind::Network; | support
| 378 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 379 | (blank) | blank separator; no runtime behavior. |
| 380 |             total_current += branch.actual_current_a; | support logic; impact depends on neighboring block and
| 381 |             let drop_v = branch.actual_current_a * (branch.resistance_ohm + ground_r + extra_r.min(1.0)); | supp
| 382 |             let target_v = (source_v - drop_v).max(0.0); | support logic; impact depends on neighboring block a
| 383 |             let tau_s = (((branch.resistance_ohm + extra_r.min(1.0)).max(0.005)) | support logic; impact depends
| 384 |                 * (branch.capacitance_nf * 1.0e-9)) | support logic; impact depends on neighboring block and dat
| 385 |                 .clamp(0.002, 0.08); | support logic; impact depends on neighboring block and data flow. |
| 386 |             branch.voltage_v += (target_v - branch.voltage_v) * (dt / tau_s).min(1.0); | support logic; impact
| 387 |             if branch.actual_current_a > branch.fuse_rating_a * 1.15 && branch.kind != ElectricalBranchKind::Ne
| 388 |                 branch.blown = true; | support logic; impact depends on neighboring block and data flow. |
| 389 |                 branch.energized = false; | support logic; impact depends on neighboring block and data flow. |
| 390 |                 branch.actual_current_a = 0.0; | support logic; impact depends on neighboring block and data flo
| 391 |                 branch.voltage_v = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 392 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 393 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 394 | (blank) | blank separator; no runtime behavior. |
| 395 |         self.snapshot.total_current_a = total_current; | support logic; impact depends on neighboring block and
```

```
| 396 |         self.snapshot.ground_drop_v = total_current * ground_r; | support logic; impact depends on neighboring b
| 397 |         self.snapshot.line31_v = self.snapshot.ground_drop_v; | support logic; impact depends on neighboring blo
| 398 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 399 | (blank) | blank separator; no runtime behavior. |
| 400 |     pub fn powered_sas(&self) -> Vec<u8> { | function signature/body; changes affect API behavior and control-fl
| 401 |         self.branches | support logic; impact depends on neighboring block and data flow. |
| 402 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 403 |             .filter_map(\|b\| { | support logic; impact depends on neighboring block and data flow. |
| 404 |                 if b.energized && !b.blown && b.voltage_v > 9.0 { | runtime gate; condition changes can enable/d
| 405 |                     b.target_sa | support logic; impact depends on neighboring block and data flow. |
| 406 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 407 |                     None | support logic; impact depends on neighboring block and data flow. |
| 408 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 409 |             }) | support logic; impact depends on neighboring block and data flow. |
| 410 |             .collect() | support logic; impact depends on neighboring block and data flow. |
| 411 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 412 | (blank) | blank separator; no runtime behavior. |
| 413 |     pub fn branch(&self, branch_id: &str) -> Option<&ElectricalBranch> { | function signature/body; changes affe
| 414 |         self.branches.iter().find(\|b\| b.id == branch_id) | support logic; impact depends on neighboring block
| 415 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 416 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 417 | (blank) | blank separator; no runtime behavior. |
| 418 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 419 | mod tests { | module declaration; changing can break namespace graph. |
| 420 |     use super::{ElectricalControlPoint, ElectricalFaultMode, ElectricalLoadRequest, ElectricalSystem}; | import
| 421 |     use crate::IgnitionState; | import wiring; changing path/name can break compilation and module linkage. |
| 422 | (blank) | blank separator; no runtime behavior. |
| 423 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 424 |     fn ignition_relay_controls_keyed_ecu_power() { | function signature/body; changes affect API behavior and co
| 425 |         let mut es = ElectricalSystem::new(); | support logic; impact depends on neighboring block and data flow
| 426 |         let loads = ElectricalLoadRequest { | support logic; impact depends on neighboring block and data flow.
| 427 |             ecm_supply_a: 6.0, | support logic; impact depends on neighboring block and data flow. |
| 428 |             tcm_supply_a: 3.0, | support logic; impact depends on neighboring block and data flow. |
| 429 |             abs_supply_a: 4.0, | support logic; impact depends on neighboring block and data flow. |
| 430 |             hcm_supply_a: 3.0, | support logic; impact depends on neighboring block and data flow. |
| 431 |             bcm_supply_a: 8.0, | support logic; impact depends on neighboring block and data flow. |
| 432 |             icm_supply_a: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 433 |             ..Default::default() | support logic; impact depends on neighboring block and data flow. |
| 434 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 435 |         es.tick(IgnitionState::On, 12.6, 14.2, &loads, &[], 0.016); | support logic; impact depends on neighbori
| 436 |         assert!(es.powered_sas().contains(&crate::j1939::addr::ECM_1)); | support logic; impact depends on neigh
| 437 | (blank) | blank separator; no runtime behavior. |
| 438 |         es.set_control_fault(ElectricalControlPoint::IgnitionRelay, ElectricalFaultMode::OpenCircuit); | support
| 439 |         es.tick(IgnitionState::On, 12.6, 14.2, &loads, &[], 0.016); | support logic; impact depends on neighbori
| 440 |         assert!(!es.powered_sas().contains(&crate::j1939::addr::ECM_1)); | support logic; impact depends on neig
| 441 |         assert!(es.powered_sas().contains(&crate::j1939::addr::CAB)); | support logic; impact depends on neighbo
| 442 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 443 | (blank) | blank separator; no runtime behavior. |
| 444 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 445 |     fn short_to_ground_blows_ecu_branch() { | function signature/body; changes affect API behavior and control-
| 446 |         let mut es = ElectricalSystem::new(); | support logic; impact depends on neighboring block and data flow
| 447 |         let loads = ElectricalLoadRequest { | support logic; impact depends on neighboring block and data flow.
| 448 |             ecm_supply_a: 6.0, | support logic; impact depends on neighboring block and data flow. |
| 449 |             ..Default::default() | support logic; impact depends on neighboring block and data flow. |
| 450 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 451 |         es.set_branch_fault("ECM-PWR", ElectricalFaultMode::ShortToGround); | support logic; impact depends on n
| 452 |         es.tick(IgnitionState::On, 12.4, 14.0, &loads, &[], 0.016); | support logic; impact depends on neighbori
| 453 |         let branch = es.branch("ECM-PWR").expect("branch present"); | panic boundary; edits alter crash behavior
| 454 |         assert!(branch.blown); | support logic; impact depends on neighboring block and data flow. |
| 455 |         assert!(!branch.energized); | support logic; impact depends on neighboring block and data flow. |
| 456 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 457 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/engine.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 126 | Non-empty: 113 | Symbol count: 10

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| torque_at_rpm | 2 |
```

```
| new | 1 |
| update | 1 |
| wheel_drive_force_n | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 2 | const | GEAR_RATIOS | namespace and compile-time linkage |
| 3 | const | FINAL_DRIVE | namespace and compile-time linkage |
| 4 | const | WHEEL_RADIUS_M | namespace and compile-time linkage |
| 5 | const | FUEL_TANK_L | namespace and compile-time linkage |
| 8 | struct | EngineEcu | data/schema coupling and match/serde break risk |
| 29 | impl | EngineEcu | method behavior and trait semantics |
| 30 | fn | new | API and behavior coupling to callers/implementers |
| 54 | fn | torque_at_rpm | API and behavior coupling to callers/implementers |
| 74 | fn | update | API and behavior coupling to callers/implementers |
| 119 | fn | wheel_drive_force_n | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | // ZF 8HP gear ratios + final drive | comment/documentation; changing affects understanding and intent traceabili
| 2 | pub const GEAR_RATIOS: [f64; 9] = [0.0, 4.714, 3.143, 2.106, 1.667, 1.285, 1.000, 0.839, 0.667]; | support logic;
| 3 | pub const FINAL_DRIVE: f64 = 3.06; | support logic; impact depends on neighboring block and data flow. |
| 4 | pub const WHEEL_RADIUS_M: f64 = 0.32; | support logic; impact depends on neighboring block and data flow. |
| 5 | pub const FUEL_TANK_L: f64 = 55.0; | support logic; impact depends on neighboring block and data flow. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 8 | pub struct EngineEcu { | data layout contract; field changes can break callers/serde/tests. |
| 9 |     pub rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 10 |     pub throttle_position: f64,    // 0-1 | support logic; impact depends on neighboring block and data flow. |
| 11 |     pub engine_temp: f64,          // °C | support logic; impact depends on neighboring block and data flow. |
| 12 |     pub oil_temp: f64,             // °C | support logic; impact depends on neighboring block and data flow. |
| 13 |     pub oil_pressure: f64,         // bar | support logic; impact depends on neighboring block and data flow. |
| 14 |     pub coolant_temp: f64,         // °C | support logic; impact depends on neighboring block and data flow. |
| 15 |     pub fuel_level: f64,           // 0-1 | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub fuel_consumption_lph: f64, // L/hour instantaneous | support logic; impact depends on neighboring block a
| 17 |     pub boost_pressure: f64,       // bar (turbo) | support logic; impact depends on neighboring block and data
| 18 |     pub intake_air_temp: f64,      // °C | support logic; impact depends on neighboring block and data flow. |
| 19 |     pub output_torque: f64,        // Nm | support logic; impact depends on neighboring block and data flow. |
| 20 |     pub output_power_kw: f64,      // kW | support logic; impact depends on neighboring block and data flow. |
| 21 |     pub check_engine: bool, | support logic; impact depends on neighboring block and data flow. |
| 22 |     pub engine_on: bool, | support logic; impact depends on neighboring block and data flow. |
| 23 |     pub rev_limiter_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 24 |     pub idle_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 25 |     pub redline_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 26 |     pub max_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 27 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 28 | (blank) | blank separator; no runtime behavior. |
| 29 | impl EngineEcu { | implementation binding; behavior semantics and trait conformance live here. |
| 30 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 31 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 32 |             rpm: 820.0, | support logic; impact depends on neighboring block and data flow. |
| 33 |             throttle_position: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 34 |             engine_temp: 22.0, | support logic; impact depends on neighboring block and data flow. |
| 35 |             oil_temp: 22.0, | support logic; impact depends on neighboring block and data flow. |
| 36 |             oil_pressure: 4.5, | support logic; impact depends on neighboring block and data flow. |
| 37 |             coolant_temp: 22.0, | support logic; impact depends on neighboring block and data flow. |
| 38 |             fuel_level: 0.85, | support logic; impact depends on neighboring block and data flow. |
| 39 |             fuel_consumption_lph: 0.6, | support logic; impact depends on neighboring block and data flow. |
| 40 |             boost_pressure: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 41 |             intake_air_temp: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 42 |             output_torque: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 43 |             output_power_kw: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 44 |             check_engine: false, | support logic; impact depends on neighboring block and data flow. |
| 45 |             engine_on: true, | support logic; impact depends on neighboring block and data flow. |
| 46 |             rev_limiter_active: false, | support logic; impact depends on neighboring block and data flow. |
| 47 |             idle_rpm: 820.0, | support logic; impact depends on neighboring block and data flow. |
| 48 |             redline_rpm: 6800.0, | support logic; impact depends on neighboring block and data flow. |
| 49 |             max_rpm: 7200.0, | support logic; impact depends on neighboring block and data flow. |
| 50 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 51 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 | (blank) | blank separator; no runtime behavior. |
```

```
| 53 |     /// Torque curve: 2.0L turbo, ~250 Nm peak at 2000-4500 RPM | comment/documentation; changing affects unders
| 54 |     fn torque_at_rpm(&self, rpm: f64, throttle: f64) -> f64 { | function signature/body; changes affect API beha
| 55 |         if rpm < 500.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 56 |             return 0.0; | support logic; impact depends on neighboring block and data flow. |
| 57 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 58 |         // Flat torque plateau between 2000-4500 RPM (turbocharged) | comment/documentation; changing affects un
| 59 |         let peak = 250.0; | support logic; impact depends on neighboring block and data flow. |
| 60 |         let t = if rpm < 1500.0 { | support logic; impact depends on neighboring block and data flow. |
| 61 |             rpm / 1500.0 * 0.7 | support logic; impact depends on neighboring block and data flow. |
| 62 |         } else if rpm < 2000.0 { | support logic; impact depends on neighboring block and data flow. |
| 63 |             0.7 + (rpm - 1500.0) / 500.0 * 0.3 | support logic; impact depends on neighboring block and data flo
| 64 |         } else if rpm < 4500.0 { | support logic; impact depends on neighboring block and data flow. |
| 65 |             1.0 | support logic; impact depends on neighboring block and data flow. |
| 66 |         } else if rpm < 6800.0 { | support logic; impact depends on neighboring block and data flow. |
| 67 |             1.0 - (rpm - 4500.0) / 2300.0 * 0.6 | support logic; impact depends on neighboring block and data fl
| 68 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 69 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 70 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 |         peak * t * throttle | support logic; impact depends on neighboring block and data flow. |
| 72 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 73 | (blank) | blank separator; no runtime behavior. |
| 74 |     pub fn update(&mut self, throttle: f64, gear: u8, vehicle_speed_kmh: f64, dt: f64) { | function signature/bo
| 75 |         self.throttle_position = throttle.clamp(0.0, 1.0); | support logic; impact depends on neighboring block a
| 76 |         self.rev_limiter_active = self.rpm >= self.redline_rpm; | support logic; impact depends on neighboring b
| 77 | (blank) | blank separator; no runtime behavior. |
| 78 |         let effective_throttle = if self.rev_limiter_active { | support logic; impact depends on neighboring blo
| 79 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 80 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 81 |             throttle | support logic; impact depends on neighboring block and data flow. |
| 82 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 83 | (blank) | blank separator; no runtime behavior. |
| 84 |         // Compute RPM from wheel speed × gear ratio | comment/documentation; changing affects understanding and
| 85 |         let target_rpm = if gear > 0 && gear <= 8 { | support logic; impact depends on neighboring block and data
| 86 |             let wheel_rps = (vehicle_speed_kmh / 3.6) / WHEEL_RADIUS_M; | support logic; impact depends on neigh
| 87 |             (wheel_rps * 60.0 * GEAR_RATIOS[gear as usize] * FINAL_DRIVE).max(self.idle_rpm) | support logic; im
| 88 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 89 |             self.idle_rpm + effective_throttle * 1500.0 | support logic; impact depends on neighboring block and
| 90 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 91 | (blank) | blank separator; no runtime behavior. |
| 92 |         self.rpm += (target_rpm - self.rpm) * (dt / 0.08).min(1.0); | support logic; impact depends on neighbori
| 93 |         self.rpm = self.rpm.clamp(self.idle_rpm * 0.9, self.max_rpm); | support logic; impact depends on neighbo
| 94 | (blank) | blank separator; no runtime behavior. |
| 95 |         self.output_torque = self.torque_at_rpm(self.rpm, effective_throttle); | support logic; impact depends o
| 96 |         self.output_power_kw = self.output_torque * self.rpm * std::f64::consts::PI / 30.0 / 1000.0; | support l
| 97 | (blank) | blank separator; no runtime behavior. |
| 98 |         // Turbo boost builds with RPM + throttle | comment/documentation; changing affects understanding and in
| 99 |         let boost_target = | support logic; impact depends on neighboring block and data flow. |
| 100 |             (effective_throttle * 1.6 * ((self.rpm - 1500.0) / 1500.0).clamp(0.0, 1.0)).max(0.0); | support log
| 101 |         self.boost_pressure += (boost_target - self.boost_pressure) * dt * 3.0; | support logic; impact depends
| 102 | (blank) | blank separator; no runtime behavior. |
| 103 |         // Thermal model | comment/documentation; changing affects understanding and intent traceability. |
| 104 |         let temp_target = 92.0 + effective_throttle * 10.0; | support logic; impact depends on neighboring bloc
| 105 |         self.engine_temp += (temp_target - self.engine_temp) * dt * 0.015; | support logic; impact depends on n
| 106 |         self.coolant_temp = self.engine_temp; | support logic; impact depends on neighboring block and data flo
| 107 |         self.oil_temp = self.engine_temp + 8.0; | support logic; impact depends on neighboring block and data f
| 108 | (blank) | blank separator; no runtime behavior. |
| 109 |         // Oil pressure drops slightly at idle, rises with RPM | comment/documentation; changing affects unders
| 110 |         self.oil_pressure = 1.2 + (self.rpm / self.max_rpm) * 4.0; | support logic; impact depends on neighbori
| 111 | (blank) | blank separator; no runtime behavior. |
| 112 |         // Fuel consumption: idle ~0.6 L/h, WOT at high RPM ~18 L/h | comment/documentation; changing affects u
| 113 |         self.fuel_consumption_lph = 0.6 + effective_throttle * 17.0 * (self.rpm / 4000.0).min(1.0); | support l
| 114 |         let consumed = self.fuel_consumption_lph * dt / 3600.0; | support logic; impact depends on neighboring
| 115 |         self.fuel_level = (self.fuel_level - consumed / FUEL_TANK_L).max(0.0); | support logic; impact depends
| 116 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 117 | (blank) | blank separator; no runtime behavior. |
| 118 |     /// Torque available at wheel (through drivetrain, at current RPM) | comment/documentation; changing affects
| 119 |     pub fn wheel_drive_force_n(&self, gear: u8) -> f64 { | function signature/body; changes affect API behavior
| 120 |         if gear == 0 \|\| gear > 8 { | runtime gate; condition changes can enable/disable critical paths. |
| 121 |             return 0.0; | support logic; impact depends on neighboring block and data flow. |
| 122 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 123 |         let wheel_torque = self.output_torque * GEAR_RATIOS[gear as usize] * FINAL_DRIVE; | support logic; impa
| 124 |         wheel_torque / WHEEL_RADIUS_M | support logic; impact depends on neighboring block and data flow. |
| 125 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 126 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/gps.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 367 | Non-empty: 335 | Symbol count: 20

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| noise | 5 |
| deg_to_nmea | 5 |
| new | 4 |
| nmea_checksum | 4 |
| fmt | 3 |
| len | 3 |
| init_satellites | 2 |
| build_gga | 2 |
| build_rmc | 2 |
| build_gsa | 2 |
| default | 1 |
| push | 1 |
| update | 1 |
| position_string | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 10 | enum | GpsFixQuality | data/schema coupling and match/serde break risk |
| 20 | impl | std::fmt::Display for GpsFixQuality | method behavior and trait semantics |
| 21 | fn | fmt | API and behavior coupling to callers/implementers |
| 36 | struct | Satellite | data/schema coupling and match/serde break risk |
| 46 | enum | Constellation | data/schema coupling and match/serde break risk |
| 54 | impl | std::fmt::Display for Constellation | method behavior and trait semantics |
| 55 | fn | fmt | API and behavior coupling to callers/implementers |
| 68 | struct | GpsModule | data/schema coupling and match/serde break risk |
| 115 | impl | Default for GpsModule | method behavior and trait semantics |
| 116 | fn | default | API and behavior coupling to callers/implementers |
| 121 | impl | GpsModule | method behavior and trait semantics |
| 122 | fn | new | API and behavior coupling to callers/implementers |
| 161 | fn | init_satellites | API and behavior coupling to callers/implementers |
| 199 | fn | update | API and behavior coupling to callers/implementers |
| 290 | fn | build_gga | API and behavior coupling to callers/implementers |
| 310 | fn | build_rmc | API and behavior coupling to callers/implementers |
| 328 | fn | build_gsa | API and behavior coupling to callers/implementers |
| 348 | fn | deg_to_nmea | API and behavior coupling to callers/implementers |
| 356 | fn | nmea_checksum | API and behavior coupling to callers/implementers |
| 360 | fn | position_string | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! GPS/GNSS Module — Full simulation of a u-blox ZED-F9P grade receiver. | comment/documentation; changing affec
| 2 | //! Outputs real NMEA 0183 sentences (GGA, RMC, GSA, GSV) with valid checksums. | comment/documentation; changing
| 3 | //! Simulates: satellite constellation, atmospheric delays, multipath, SBAS correction. | comment/documentation; c
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | use std::f64::consts::PI; | import wiring; changing path/name can break compilation and module linkage. |
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | // ■■ Fix Quality ■■■
```

```
| 21 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 22 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 23 |             GpsFixQuality::NoFix => write!(f, "NO FIX"), | protocol or I/O critical; changes may impact external
| 24 |             GpsFixQuality::SpsFix => write!(f, "SPS (±3m) "), | protocol or I/O critical; changes may impact exte
| 25 |             GpsFixQuality::DgpsFix => write!(f, "DGPS(±1m) "), | protocol or I/O critical; changes may impact ext
| 26 |             GpsFixQuality::RtkFix => write!(f, "RTK (±2cm)"), | protocol or I/O critical; changes may impact exte
| 27 |             GpsFixQuality::FloatRtk => write!(f, "RTK-Float "), | protocol or I/O critical; changes may impact ex
| 28 |             GpsFixQuality::Dead => write!(f, "DR        "), | protocol or I/O critical; changes may impact extern
| 29 |             _ => write!(f, "PPS       "), | protocol or I/O critical; changes may impact external integration co
| 30 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 31 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 32 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 | // ■■ Satellite ■■■
```

```
| 97 |     pub utc_date: String, // "DDMMYY" | support logic; impact depends on neighboring block and data flow. |
| 98 |     pub gps_week: u16, | support logic; impact depends on neighboring block and data flow. |
| 99 |     pub time_of_week_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 100 | (blank) | blank separator; no runtime behavior. |
| 101 |     // ■ Dead Reckoning ■■■
```

```
| 173 |             (22, 38.0, 265.0, 39.0, Constellation::Gps), | support logic; impact depends on neighboring block ar
| 174 |             (28, 85.0, 330.0, 48.0, Constellation::Gps), | support logic; impact depends on neighboring block ar
| 175 |             (65, 42.0, 95.0, 37.0, Constellation::Glonass), | support logic; impact depends on neighboring bloc
| 176 |             (66, 29.0, 175.0, 33.0, Constellation::Glonass), | support logic; impact depends on neighboring bloc
| 177 |             (67, 58.0, 250.0, 43.0, Constellation::Glonass), | support logic; impact depends on neighboring bloc
| 178 |             (68, 15.0, 310.0, 24.0, Constellation::Glonass), | support logic; impact depends on neighboring bloc
| 179 |             (11 + 64, 47.0, 60.0, 40.0, Constellation::Galileo), | support logic; impact depends on neighboring
| 180 |             (12 + 64, 33.0, 220.0, 36.0, Constellation::Galileo), | support logic; impact depends on neighboring
| 181 |             (13 + 64, 68.0, 145.0, 46.0, Constellation::Galileo), | support logic; impact depends on neighboring
| 182 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 183 |         for (prn, elev, azimuth, snr, constellation) in sat_data.iter() { | iteration logic; may alter timing, t
| 184 |             self.satellites.push(Satellite { | support logic; impact depends on neighboring block and data flow
| 185 |                 prn: *prn, | support logic; impact depends on neighboring block and data flow. |
| 186 |                 elevation: *elev, | support logic; impact depends on neighboring block and data flow. |
| 187 |                 azimuth: *azimuth, | support logic; impact depends on neighboring block and data flow. |
| 188 |                 snr: *snr, | support logic; impact depends on neighboring block and data flow. |
| 189 |                 used: *elev > 10.0, | support logic; impact depends on neighboring block and data flow. |
| 190 |                 constellation: *constellation, | support logic; impact depends on neighboring block and data flo
| 191 |             }); | support logic; impact depends on neighboring block and data flow. |
| 192 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 193 |         self.sats_in_view = self.satellites.len() as u8; | support logic; impact depends on neighboring block ar
| 194 |         self.sats_used = self.satellites.iter().filter(\|s\| s.used).count() as u8; | support logic; impact depe
| 195 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 196 | (blank) | blank separator; no runtime behavior. |
| 197 |     /// Main update — call every simulation step. | comment/documentation; changing affects understanding and in
| 198 |     /// `speed_kmh` and `heading_deg` come from vehicle dynamics. | comment/documentation; changing affects unde
| 199 |     pub fn update(&mut self, speed_kmh: f64, heading_deg: f64, dt: f64) { | function signature/body; changes af
| 200 |         self.update_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 201 | (blank) | blank separator; no runtime behavior. |
| 202 |         // ■ Update position from vehicle speed ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentation;
| 203 |         let speed_ms = speed_kmh / 3.6; | support logic; impact depends on neighboring block and data flow. |
| 204 |         let head_rad = heading_deg * PI / 180.0; | support logic; impact depends on neighboring block and data f
| 205 |         self.pos_x += speed_ms * head_rad.sin() * dt; | support logic; impact depends on neighboring block and d
| 206 |         self.pos_y += speed_ms * head_rad.cos() * dt; | support logic; impact depends on neighboring block and d
| 207 | (blank) | blank separator; no runtime behavior. |
| 208 |         // ■ Convert local ENU to lat/lon ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentation
| 209 |         // 1° latitude  ≈ 111,320 m | comment/documentation; changing affects understanding and intent traceabil
| 210 |         // 1° longitude ≈ 111,320 × cos(lat) m | comment/documentation; changing affects understanding and inten
| 211 |         let metres_per_deg_lat = 111_320.0; | support logic; impact depends on neighboring block and data flow.
| 212 |         let metres_per_deg_lon = 111_320.0 * (self.latitude_deg * PI / 180.0).cos(); | support logic; impact dep
| 213 |         self.latitude_deg += (self.pos_y / metres_per_deg_lat) * 0.001 * dt; // slow drift | support logic; impa
| 214 |         self.longitude_deg += (self.pos_x / metres_per_deg_lon) * 0.001 * dt; | support logic; impact depends on
| 215 |         // Actual integration — more accurate | comment/documentation; changing affects understanding and intent
| 216 |         let d_lat = (speed_ms * head_rad.cos() * dt) / metres_per_deg_lat; | support logic; impact depends on ne
| 217 |         let d_lon = (speed_ms * head_rad.sin() * dt) / metres_per_deg_lon; | support logic; impact depends on ne
| 218 |         self.latitude_deg += d_lat; | support logic; impact depends on neighboring block and data flow. |
| 219 |         self.longitude_deg += d_lon; | support logic; impact depends on neighboring block and data flow. |
| 220 | (blank) | blank separator; no runtime behavior. |
| 221 |         self.speed_kmh = speed_kmh; | support logic; impact depends on neighboring block and data flow. |
| 222 |         self.speed_knots = speed_kmh * 0.539957; | support logic; impact depends on neighboring block and data f
| 223 |         self.course_deg = heading_deg; | support logic; impact depends on neighboring block and data flow. |
| 224 | (blank) | blank separator; no runtime behavior. |
| 225 |         // ■ Add GPS noise (pseudo-random, position-dependent) ■■■■■■■■■■■■■■■■ | comment/documentation; cha
| 226 |         self.noise_seed += dt * 7.3; | support logic; impact depends on neighboring block and data flow. |
| 227 |         let noise = \|s: f64\| ((s * 127.1 + 311.7).sin() * 43758.5 % 1.0 - 0.5) * 2.0; | support logic; impact
| 228 |         let pos_noise_m = match self.fix_quality { | support logic; impact depends on neighboring block and data
| 229 |             GpsFixQuality::RtkFix => 0.02, | support logic; impact depends on neighboring block and data flow.
| 230 |             GpsFixQuality::DgpsFix => 0.5, | support logic; impact depends on neighboring block and data flow.
| 231 |             GpsFixQuality::SpsFix => 2.5, | support logic; impact depends on neighboring block and data flow. |
| 232 |             GpsFixQuality::Dead => self.dr_error_m, | support logic; impact depends on neighboring block and da
| 233 |             _ => 5.0, | support logic; impact depends on neighboring block and data flow. |
| 234 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 235 |         let noise_lat = noise(self.noise_seed) * pos_noise_m / metres_per_deg_lat; | support logic; impact depe
| 236 |         let noise_lon = noise(self.noise_seed + 1.5) * pos_noise_m / metres_per_deg_lon; | support logic; impac
| 237 |         self.hacc_m = pos_noise_m * 1.5; | support logic; impact depends on neighboring block and data flow. |
| 238 |         self.vacc_m = pos_noise_m * 2.5; | support logic; impact depends on neighboring block and data flow. |
| 239 | (blank) | blank separator; no runtime behavior. |
| 240 |         // ■ Satellite dynamics (elevation/SNR vary slowly) ■■■■■■■■■■■■■■■■■■■■ | comment/documentation; ch
| 241 |         let t = self.noise_seed; | support logic; impact depends on neighboring block and data flow. |
| 242 |         for (i, sat) in self.satellites.iter_mut().enumerate() { | iteration logic; may alter timing, throughpu
| 243 |             let drift = (t * 0.01 + i as f64 * 0.7).sin() * 0.3; | support logic; impact depends on neighboring
| 244 |             sat.elevation = (sat.elevation + drift * dt * 0.1).clamp(5.0, 90.0); | support logic; impact depends
| 245 |             let snr_noise = noise(t + i as f64 * 3.1) * 0.5; | support logic; impact depends on neighboring bloc
| 246 |             sat.snr = (sat.snr + snr_noise * dt * 5.0).clamp(10.0, 52.0); | support logic; impact depends on ne
| 247 |             sat.used = sat.elevation > 10.0 && sat.snr > 18.0; | support logic; impact depends on neighboring b
| 248 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 249 |         self.sats_used = self.satellites.iter().filter(\|s\| s.used).count() as u8; | support logic; impact depe
| 250 |         self.sats_in_view = self.satellites.len() as u8; | support logic; impact depends on neighboring block an
| 251 | (blank) | blank separator; no runtime behavior. |
| 252 |         // ■ DOP calculation ━━━
```

```
| 325 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 326 | (blank) | blank separator; no runtime behavior. |
| 327 |     /// GSA — GNSS DOP and Active Satellites | comment/documentation; changing affects understanding and intent
| 328 |     fn build_gsa(&self) -> String { | function signature/body; changes affect API behavior and control-flow. |
| 329 |         let used: String = self | support logic; impact depends on neighboring block and data flow. |
| 330 |             .satellites | support logic; impact depends on neighboring block and data flow. |
| 331 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 332 |             .filter(\|s\| s.used) | support logic; impact depends on neighboring block and data flow. |
| 333 |             .take(12) | support logic; impact depends on neighboring block and data flow. |
| 334 |             .map(\|s\| format!("{:02}", s.prn)) | support logic; impact depends on neighboring block and data f
| 335 |             .collect::<Vec<_>>() | support logic; impact depends on neighboring block and data flow. |
| 336 |             .join(","); | support logic; impact depends on neighboring block and data flow. |
| 337 |         let body = format!( | support logic; impact depends on neighboring block and data flow. |
| 338 |             "GPGSA,A,3,{},{},{:.1},{:.1},{:.1}", | support logic; impact depends on neighboring block and data
| 339 |             used, | support logic; impact depends on neighboring block and data flow. |
| 340 |             ",".repeat(12usize.saturating_sub(self.sats_used as usize)), | support logic; impact depends on nei
| 341 |             self.pdop, | support logic; impact depends on neighboring block and data flow. |
| 342 |             self.hdop, | support logic; impact depends on neighboring block and data flow. |
| 343 |             self.vdop | support logic; impact depends on neighboring block and data flow. |
| 344 |         ); | support logic; impact depends on neighboring block and data flow. |
| 345 |         format!("${}*{:02X}\r\n", body, Self::nmea_checksum(&body)) | support logic; impact depends on neighbor
| 346 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 347 | (blank) | blank separator; no runtime behavior. |
| 348 |     fn deg_to_nmea(deg: f64, pos: char, neg: char) -> (f64, f64, char) { | function signature/body; changes aff
| 349 |         let hemi = if deg >= 0.0 { pos } else { neg }; | support logic; impact depends on neighboring block and
| 350 |         let abs = deg.abs(); | support logic; impact depends on neighboring block and data flow. |
| 351 |         let d = abs.floor(); | support logic; impact depends on neighboring block and data flow. |
| 352 |         let m = (abs - d) * 60.0; | support logic; impact depends on neighboring block and data flow. |
| 353 |         (d, d * 100.0 + m, hemi) | support logic; impact depends on neighboring block and data flow. |
| 354 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 355 | (blank) | blank separator; no runtime behavior. |
| 356 |     fn nmea_checksum(sentence: &str) -> u8 { | function signature/body; changes affect API behavior and control
| 357 |         sentence.bytes().fold(0u8, \|acc, b\| acc ^ b) | support logic; impact depends on neighboring block and
| 358 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 359 | (blank) | blank separator; no runtime behavior. |
| 360 |     pub fn position_string(&self) -> String { | function signature/body; changes affect API behavior and contro
| 361 |         let lat_d = self.latitude_deg.abs(); | support logic; impact depends on neighboring block and data flow
| 362 |         let lat_h = if self.latitude_deg >= 0.0 { 'N' } else { 'S' }; | support logic; impact depends on neighb
| 363 |         let lon_d = self.longitude_deg.abs(); | support logic; impact depends on neighboring block and data flo
| 364 |         let lon_h = if self.longitude_deg >= 0.0 { 'E' } else { 'W' }; | support logic; impact depends on neigh
| 365 |         format!("{:.6}°{}  {:.6}°{}", lat_d, lat_h, lon_d, lon_h) | support logic; impact depends on neighborin
| 366 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 367 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/implement.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 505 | Non-empty: 442 | Symbol count: 36

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 8 |
| fmt | 4 |
| update | 3 |
| target_rpm | 2 |
| update_pto | 2 |
| update_hitch | 2 |
| update_aux_hydraulics | 2 |
| push | 2 |
| required_engine_rpm | 1 |
| pto_torque_demand | 1 |
| generate_j1939_frames | 1 |
| pto | 1 |
| from_raw | 1 |
| build_id | 1 |
| default | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
```

```
| 15 | const | HITCH_MAX_RATE | namespace and compile-time linkage |
| 17 | const | PTO_OVERLOAD_SLIP | namespace and compile-time linkage |
| 19 | const | PTO_RECOVER_SLIP | namespace and compile-time linkage |
| 21 | const | AUX_MAX_FLOW | namespace and compile-time linkage |
| 23 | const | AUX_RELIEF_BAR | namespace and compile-time linkage |
| 25 | const | HITCH_SENSOR_BW | namespace and compile-time linkage |
| 31 | enum | PtoMode | data/schema coupling and match/serde break risk |
| 39 | impl | PtoMode | method behavior and trait semantics |
| 41 | fn | target_rpm | API and behavior coupling to callers/implementers |
| 50 | fn | required_engine_rpm | API and behavior coupling to callers/implementers |
| 61 | impl | fmt::Display for PtoMode | method behavior and trait semantics |
| 62 | fn | fmt | API and behavior coupling to callers/implementers |
| 77 | enum | HitchMode | data/schema coupling and match/serde break risk |
| 88 | impl | fmt::Display for HitchMode | method behavior and trait semantics |
| 89 | fn | fmt | API and behavior coupling to callers/implementers |
| 103 | enum | ImplementType | data/schema coupling and match/serde break risk |
| 114 | impl | fmt::Display for ImplementType | method behavior and trait semantics |
| 115 | fn | fmt | API and behavior coupling to callers/implementers |
| 133 | enum | ValveDirection | data/schema coupling and match/serde break risk |
| 139 | impl | fmt::Display for ValveDirection | method behavior and trait semantics |
| 140 | fn | fmt | API and behavior coupling to callers/implementers |
| 153 | struct | HydraulicBank | data/schema coupling and match/serde break risk |
| 163 | impl | HydraulicBank | method behavior and trait semantics |
| 164 | fn | new | API and behavior coupling to callers/implementers |
| 176 | fn | update | API and behavior coupling to callers/implementers |
| 193 | struct | ImplementControl | data/schema coupling and match/serde break risk |
| 234 | impl | ImplementControl | method behavior and trait semantics |
| 235 | fn | new | API and behavior coupling to callers/implementers |
| 278 | fn | update | API and behavior coupling to callers/implementers |
| 286 | fn | update_pto | API and behavior coupling to callers/implementers |
| 353 | fn | update_hitch | API and behavior coupling to callers/implementers |
| 405 | fn | update_aux_hydraulics | API and behavior coupling to callers/implementers |
| 448 | fn | pto_torque_demand | API and behavior coupling to callers/implementers |
| 463 | fn | generate_j1939_frames | API and behavior coupling to callers/implementers |
| 501 | impl | Default for ImplementControl | method behavior and trait semantics |
| 502 | fn | default | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! ISOBUS / ISO 11783-7 Implement Controller | comment/documentation; changing affects understanding and intent t
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! Covers: Rear & Front PTO, 3-Point Hitch (position / draft / mixed / float), | comment/documentation; changing
| 4 | //! four Auxiliary Hydraulic valve banks, loader hydraulics, and Task Controller | comment/documentation; changing
| 5 | //! interface.  Emits J1939 PGN 65093 (PTO) and PGN 65091 (Hitch) frames. | comment/documentation; changing affec
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | use crate::j1939::addr; | import wiring; changing path/name can break compilation and module linkage. |
| 8 | use crate::j1939::pgn; | import wiring; changing path/name can break compilation and module linkage. |
| 9 | use crate::j1939::{Builder, J1939Frame}; | import wiring; changing path/name can break compilation and module lin
| 10 | use std::fmt; | import wiring; changing path/name can break compilation and module linkage. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | // ■■ Module-level constants ■■■
```

```
| 36 |     Economy1000, // 1000 RPM at reduced engine speed | support logic; impact depends on neighboring block and dat
| 37 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | (blank) | blank separator; no runtime behavior. |
| 39 | impl PtoMode { | implementation binding; behavior semantics and trait conformance live here. |
| 40 |     /// Target PTO shaft output speed [RPM] | comment/documentation; changing affects understanding and intent tr
| 41 |     pub fn target_rpm(self) -> f64 { | function signature/body; changes affect API behavior and control-flow. |
| 42 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 43 |             PtoMode::Off => 0.0, | support logic; impact depends on neighboring block and data flow. |
| 44 |             PtoMode::Std540 \| PtoMode::Economy540 => 540.0, | support logic; impact depends on neighboring bloc
| 45 |             PtoMode::Std1000 \| PtoMode::Economy1000 => 1000.0, | support logic; impact depends on neighboring b
| 46 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 | (blank) | blank separator; no runtime behavior. |
| 49 |     /// Nominal engine RPM required to achieve the target PTO speed | comment/documentation; changing affects und
| 50 |     pub fn required_engine_rpm(self) -> f64 { | function signature/body; changes affect API behavior and control-
| 51 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 52 |             PtoMode::Off => 800.0, | support logic; impact depends on neighboring block and data flow. |
| 53 |             PtoMode::Std540 => 2100.0, | support logic; impact depends on neighboring block and data flow. |
| 54 |             PtoMode::Economy540 => 1450.0, | support logic; impact depends on neighboring block and data flow. |
| 55 |             PtoMode::Std1000 => 2100.0, | support logic; impact depends on neighboring block and data flow. |
| 56 |             PtoMode::Economy1000 => 1700.0, | support logic; impact depends on neighboring block and data flow.
| 57 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 58 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 59 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 | impl fmt::Display for PtoMode { | implementation binding; behavior semantics and trait conformance live here. |
| 62 |     fn fmt(&self, f: &mut fmt::Formatter<'_>) -> fmt::Result { | function signature/body; changes affect API beha
| 63 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 64 |             PtoMode::Off => write!(f, "OFF      "), | protocol or I/O critical; changes may impact external integ
| 65 |             PtoMode::Std540 => write!(f, "540 STD  "), | protocol or I/O critical; changes may impact external in
| 66 |             PtoMode::Economy540 => write!(f, "540 ECO  "), | protocol or I/O critical; changes may impact externa
| 67 |             PtoMode::Std1000 => write!(f, "1000 STD "), | protocol or I/O critical; changes may impact external i
| 68 |             PtoMode::Economy1000 => write!(f, "1000 ECO "), | protocol or I/O critical; changes may impact extern
| 69 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 70 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 72 | (blank) | blank separator; no runtime behavior. |
| 73 | // ■■ HitchMode ■■■
```

```
| 112 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 113 | (blank) | blank separator; no runtime behavior. |
| 114 | impl fmt::Display for ImplementType { | implementation binding; behavior semantics and trait conformance live he
| 115 |     fn fmt(&self, f: &mut fmt::Formatter<'_>) -> fmt::Result { | function signature/body; changes affect API bel
| 116 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 117 |             ImplementType::Plow => write!(f, "Plow      "), | protocol or I/O critical; changes may impact exter
| 118 |             ImplementType::Disc => write!(f, "Disc      "), | protocol or I/O critical; changes may impact exter
| 119 |             ImplementType::Cultivator => write!(f, "Cultivator"), | protocol or I/O critical; changes may impact
| 120 |             ImplementType::Planter => write!(f, "Planter   "), | protocol or I/O critical; changes may impact ex
| 121 |             ImplementType::Sprayer => write!(f, "Sprayer   "), | protocol or I/O critical; changes may impact ex
| 122 |             ImplementType::Harvester => write!(f, "Harvester "), | protocol or I/O critical; changes may impact
| 123 |             ImplementType::Loader => write!(f, "Loader    "), | protocol or I/O critical; changes may impact ext
| 124 |             ImplementType::None => write!(f, "None      "), | protocol or I/O critical; changes may impact exter
| 125 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 126 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 127 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 128 | (blank) | blank separator; no runtime behavior. |
| 129 | // ■■ ValveDirection ■■■
```

```
| 188 | (blank) | blank separator; no runtime behavior. |
| 189 | // ■■ ImplementControl ■■■
```

```
| 264 | (blank) | blank separator; no runtime behavior. |
| 265 |             implement_attached: None, | support logic; impact depends on neighboring block and data flow. |
| 266 |             implement_width_m: 3.0, | support logic; impact depends on neighboring block and data flow. |
| 267 |             implement_working_depth_cm: 0.0, | support logic; impact depends on neighboring block and data flow
| 268 |             isobus_connected: false, | support logic; impact depends on neighboring block and data flow. |
| 269 |             task_controller_active: false, | support logic; impact depends on neighboring block and data flow.
| 270 | (blank) | blank separator; no runtime behavior. |
| 271 |             overload_protection_active: false, | support logic; impact depends on neighboring block and data flo
| 272 |             hitch_draft_target_kn: 8.0, | support logic; impact depends on neighboring block and data flow. |
| 273 |             pto_spin_up_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 274 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 275 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 276 | (blank) | blank separator; no runtime behavior. |
| 277 |     /// Advance simulation by `dt` seconds. | comment/documentation; changing affects understanding and intent t
| 278 |     pub fn update(&mut self, throttle: f64, ground_speed_kmh: f64, dt: f64) { | function signature/body; changes
| 279 |         self.update_pto(throttle, dt); | support logic; impact depends on neighboring block and data flow. |
| 280 |         self.update_hitch(ground_speed_kmh, dt); | support logic; impact depends on neighboring block and data f
| 281 |         self.update_aux_hydraulics(dt); | support logic; impact depends on neighboring block and data flow. |
| 282 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 283 | (blank) | blank separator; no runtime behavior. |
| 284 |     // ■■ PTO dynamics ■■■
```

```
| 340 | (blank) | blank separator; no runtime behavior. |
| 341 |         // Front PTO: driven from same shaft at a fixed reduction ratio | comment/documentation; changing affec
| 342 |         self.pto_front_rpm = if self.pto_front_enabled { | support logic; impact depends on neighboring block a
| 343 |             self.pto_rear_rpm * 0.72 | support logic; impact depends on neighboring block and data flow. |
| 344 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 345 |             (self.pto_front_rpm - 12.0 * dt).max(0.0) | support logic; impact depends on neighboring block and
| 346 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 347 | (blank) | blank separator; no runtime behavior. |
| 348 |         self.pto_spin_up_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 349 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 350 | (blank) | blank separator; no runtime behavior. |
| 351 |     // ■■ 3-Point Hitch ■■■
```

```
| 416 |                 ValveDirection::Retract => -18.0 * dt, | support logic; impact depends on neighboring block and
| 417 |                 ValveDirection::Neutral => 0.0, | support logic; impact depends on neighboring block and data f
| 418 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 419 |             self.loader_lift_position_pct = | support logic; impact depends on neighboring block and data flow.
| 420 |                 (self.loader_lift_position_pct + delta).clamp(0.0, 100.0); | support logic; impact depends on n
| 421 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 422 | (blank) | blank separator; no runtime behavior. |
| 423 |         // Bank 1 → loader/bucket tilt cylinder | comment/documentation; changing affects understanding and int
| 424 |         if self.aux_banks[1].engaged { | runtime gate; condition changes can enable/disable critical paths. |
| 425 |             let delta = match self.aux_banks[1].direction { | support logic; impact depends on neighboring bloc
| 426 |                 ValveDirection::Extend => 25.0 * dt, | support logic; impact depends on neighboring block and d
| 427 |                 ValveDirection::Retract => -25.0 * dt, | support logic; impact depends on neighboring block and
| 428 |                 ValveDirection::Neutral => 0.0, | support logic; impact depends on neighboring block and data f
| 429 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 430 |             self.loader_tilt_position_pct = | support logic; impact depends on neighboring block and data flow.
| 431 |                 (self.loader_tilt_position_pct + delta).clamp(0.0, 100.0); | support logic; impact depends on n
| 432 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 433 | (blank) | blank separator; no runtime behavior. |
| 434 |         let active_p = if self.aux_banks[0].engaged \|\| self.aux_banks[1].engaged { | support logic; impact de
| 435 |             self.aux_banks[0] | support logic; impact depends on neighboring block and data flow. |
| 436 |                 .pressure_bar | support logic; impact depends on neighboring block and data flow. |
| 437 |                 .max(self.aux_banks[1].pressure_bar) | support logic; impact depends on neighboring block and d
| 438 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 439 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 440 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 441 |         // Bleed loader circuit when nothing is active | comment/documentation; changing affects understanding
| 442 |         self.loader_pressure_bar += (active_p - self.loader_pressure_bar) * (5.0 * dt).min(1.0); | support logi
| 443 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 444 | (blank) | blank separator; no runtime behavior. |
| 445 |     // ■■ Queries ■■■
```

```
| 492 |                 J1939Frame::build_id(6, pgn::HITCH, addr::HITCH, 0xFF), | support logic; impact depends on neigh
| 493 |                 &data, | support logic; impact depends on neighboring block and data flow. |
| 494 |             )); | support logic; impact depends on neighboring block and data flow. |
| 495 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 496 | (blank) | blank separator; no runtime behavior. |
| 497 |         frames | support logic; impact depends on neighboring block and data flow. |
| 498 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 499 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 500 | (blank) | blank separator; no runtime behavior. |
| 501 | impl Default for ImplementControl { | implementation binding; behavior semantics and trait conformance live here
| 502 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 503 |         ImplementControl::new() | support logic; impact depends on neighboring block and data flow. |
| 504 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 505 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/imu.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 347 | Non-empty: 305 | Symbol count: 20

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| normalise | 3 |
| identity | 2 |
| to_euler_deg | 2 |
| new | 2 |
| ncyl | 2 |
| madgwick_update | 2 |
| default | 1 |
| update | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | const | ACCEL_NOISE | namespace and compile-time linkage |
| 9 | const | GYRO_NOISE | namespace and compile-time linkage |
| 10 | const | MAG_NOISE | namespace and compile-time linkage |
| 12 | const | ACCEL_BIAS_STD | namespace and compile-time linkage |
| 14 | const | GYRO_BIAS_STD | namespace and compile-time linkage |
| 15 | const | GYRO_DRIFT | namespace and compile-time linkage |
| 16 | const | BETA | namespace and compile-time linkage |
| 20 | struct | Quaternion | data/schema coupling and match/serde break risk |
| 27 | impl | Quaternion | method behavior and trait semantics |
| 28 | fn | identity | API and behavior coupling to callers/implementers |
| 37 | fn | normalise | API and behavior coupling to callers/implementers |
| 48 | fn | to_euler_deg | API and behavior coupling to callers/implementers |
| 59 | struct | Imu | data/schema coupling and match/serde break risk |
| 108 | impl | Default for Imu | method behavior and trait semantics |
| 109 | fn | default | API and behavior coupling to callers/implementers |
| 114 | impl | Imu | method behavior and trait semantics |
| 115 | fn | new | API and behavior coupling to callers/implementers |
| 154 | fn | update | API and behavior coupling to callers/implementers |
| 245 | fn | madgwick_update | API and behavior coupling to callers/implementers |
| 344 | fn | ncyl | API and behavior coupling to callers/implementers |
```

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! IMU — 9-DOF Inertial Measurement Unit (Accelerometer + Gyroscope + Magnetometer) | comment/documentation; char
| 2 | //! Implements the Madgwick AHRS filter for attitude estimation. | comment/documentation; changing affects understa
| 3 | //! Models: noise, bias drift, temperature coefficient, vibration. | comment/documentation; changing affects under
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use std::f64::consts::PI; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | // ■■ IMU configuration ■■■
```

```
| 11 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 12 | const ACCEL_BIAS_STD: f64 = 0.01; | support logic; impact depends on neighboring block and data flow. |
| 13 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 14 | const GYRO_BIAS_STD: f64 = 0.001; | support logic; impact depends on neighboring block and data flow. |
| 15 | const GYRO_DRIFT: f64 = 5e-7; // rad/s per second (in-run bias stability) | support logic; impact depends on neig
| 16 | const BETA: f64 = 0.1; // Madgwick filter gain (0.01=slow, 0.5=fast) | support logic; impact depends on neighbori
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | // ■■ Quaternion ■■■
```

```
| 87 | (blank) | blank separator; no runtime behavior. |
| 88 |     // ■ Temperature ■■■
```

```
| 163 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 164 |         self.noise_t += dt; | support logic; impact depends on neighboring block and data flow. |
| 165 |         let n = self.noise_t; | support logic; impact depends on neighboring block and data flow. |
| 166 | (blank) | blank separator; no runtime behavior. |
| 167 |         // ■ Engine vibration model (fundamental + harmonics) ■■■■■■■■■■■■■■■■■■■ | comment/documentation; ch
| 168 |         let vibe_freq = rpm / 60.0 * (Imu::ncyl() as f64 / 2.0); // firing frequency Hz | support logic; impact
| 169 |         self.vibe_amp = (rpm / 2200.0 * 0.15).min(0.3); | support logic; impact depends on neighboring block and
| 170 |         let vibe = self.vibe_amp | support logic; impact depends on neighboring block and data flow. |
| 171 |             * ((2.0 * PI * vibe_freq * n).sin() + 0.3 * (4.0 * PI * vibe_freq * n).sin()); | support logic; impa
| 172 | (blank) | blank separator; no runtime behavior. |
| 173 |         // ■ True accelerations in body frame ■■■■■■■■■■■■■■■■■■■
```

```
| 239 |         let accel_mag = (self.accel_x.powi(2) + self.accel_y.powi(2) + self.accel_z.powi(2)).sqrt(); | support l
| 240 |         self.accel_fault = !(4.0..=25.0).contains(&accel_mag); | support logic; impact depends on neighboring bl
| 241 |         self.gyro_fault = self.gyro_z.abs() > 10.0; // >10 rad/s = sensor fault | support logic; impact depends
| 242 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 243 | (blank) | blank separator; no runtime behavior. |
| 244 |     /// Madgwick AHRS algorithm — fuses accel + gyro + mag into quaternion attitude. | comment/documentation; ch
| 245 |     fn madgwick_update(&mut self, dt: f64) { | function signature/body; changes affect API behavior and control
| 246 |         let [ax, ay, az] = self.accel_cal; | support logic; impact depends on neighboring block and data flow.
| 247 |         let [gx, gy, gz] = self.gyro_cal; | support logic; impact depends on neighboring block and data flow. |
| 248 |         let (mx, my, mz) = (self.mag_x, self.mag_y, self.mag_z); | support logic; impact depends on neighboring
| 249 | (blank) | blank separator; no runtime behavior. |
| 250 |         let q0 = self.q.w; | support logic; impact depends on neighboring block and data flow. |
| 251 |         let q1 = self.q.x; | support logic; impact depends on neighboring block and data flow. |
| 252 |         let q2 = self.q.y; | support logic; impact depends on neighboring block and data flow. |
| 253 |         let q3 = self.q.z; | support logic; impact depends on neighboring block and data flow. |
| 254 | (blank) | blank separator; no runtime behavior. |
| 255 |         // Normalise accelerometer | comment/documentation; changing affects understanding and intent traceabili
| 256 |         let an = (ax * ax + ay * ay + az * az).sqrt(); | support logic; impact depends on neighboring block and
| 257 |         if an < 1e-10 { | runtime gate; condition changes can enable/disable critical paths. |
| 258 |             return; | support logic; impact depends on neighboring block and data flow. |
| 259 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 260 |         let (ax, ay, az) = (ax / an, ay / an, az / an); | support logic; impact depends on neighboring block and
| 261 | (blank) | blank separator; no runtime behavior. |
| 262 |         // Normalise magnetometer | comment/documentation; changing affects understanding and intent traceabili
| 263 |         let mn = (mx * mx + my * my + mz * mz).sqrt(); | support logic; impact depends on neighboring block and
| 264 |         if mn < 1e-10 { | runtime gate; condition changes can enable/disable critical paths. |
| 265 |             return; | support logic; impact depends on neighboring block and data flow. |
| 266 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 267 |         let (mx, my, mz) = (mx / mn, my / mn, mz / mn); | support logic; impact depends on neighboring block and
| 268 | (blank) | blank separator; no runtime behavior. |
| 269 |         // Reference direction of earth's magnetic field | comment/documentation; changing affects understanding
| 270 |         let hx = 2.0 * mx * (0.5 - q2 * q2 - q3 * q3) | support logic; impact depends on neighboring block and
| 271 |             + 2.0 * my * (q1 * q2 - q0 * q3) | support logic; impact depends on neighboring block and data flow
| 272 |             + 2.0 * mz * (q1 * q3 + q0 * q2); | support logic; impact depends on neighboring block and data flow
| 273 |         let hy = 2.0 * mx * (q1 * q2 + q0 * q3) | support logic; impact depends on neighboring block and data fl
| 274 |             + 2.0 * my * (0.5 - q1 * q1 - q3 * q3) | support logic; impact depends on neighboring block and data
| 275 |             + 2.0 * mz * (q2 * q3 - q0 * q1); | support logic; impact depends on neighboring block and data flow
| 276 |         let hz = 2.0 * mx * (q1 * q3 - q0 * q2) | support logic; impact depends on neighboring block and data fl
| 277 |             + 2.0 * my * (q2 * q3 + q0 * q1) | support logic; impact depends on neighboring block and data flow
| 278 |             + 2.0 * mz * (0.5 - q1 * q1 - q2 * q2); | support logic; impact depends on neighboring block and da
| 279 |         let bx = (hx * hx + hy * hy).sqrt(); | support logic; impact depends on neighboring block and data flow
| 280 |         let bz = hz; | support logic; impact depends on neighboring block and data flow. |
| 281 | (blank) | blank separator; no runtime behavior. |
| 282 |         // Gradient descent objective function | comment/documentation; changing affects understanding and inter
| 283 |         let f1 = 2.0 * (q1 * q3 - q0 * q2) - ax; | support logic; impact depends on neighboring block and data
| 284 |         let f2 = 2.0 * (q0 * q1 + q2 * q3) - ay; | support logic; impact depends on neighboring block and data
| 285 |         let f3 = 1.0 - 2.0 * (q1 * q1 + q2 * q2) - az; | support logic; impact depends on neighboring block and
| 286 |         let f4 = 2.0 * bx * (0.5 - q2 * q2 - q3 * q3) + 2.0 * bz * (q1 * q3 - q0 * q2) - mx; | support logic; in
| 287 |         let f5 = 2.0 * bx * (q1 * q2 - q0 * q3) + 2.0 * bz * (q0 * q1 + q2 * q3) - my; | support logic; impact
| 288 |         let f6 = 2.0 * bx * (q0 * q2 + q1 * q3) + 2.0 * bz * (0.5 - q1 * q1 - q2 * q2) - mz; | support logic; in
| 289 | (blank) | blank separator; no runtime behavior. |
| 290 |         // Jacobian and gradient | comment/documentation; changing affects understanding and intent traceability
| 291 |         let j11 = -2.0 * q2; | support logic; impact depends on neighboring block and data flow. |
| 292 |         let j12 = 2.0 * q3; | support logic; impact depends on neighboring block and data flow. |
| 293 |         let j13 = -2.0 * q0; | support logic; impact depends on neighboring block and data flow. |
| 294 |         let j14 = 2.0 * q1; | support logic; impact depends on neighboring block and data flow. |
| 295 |         let j21 = 2.0 * q0; | support logic; impact depends on neighboring block and data flow. |
| 296 |         let j22 = 2.0 * q1; | support logic; impact depends on neighboring block and data flow. |
| 297 |         let j23 = 2.0 * q2; | support logic; impact depends on neighboring block and data flow. |
| 298 |         let j24 = 2.0 * q3; | support logic; impact depends on neighboring block and data flow. |
| 299 |         let j31 = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 300 |         let j32 = -4.0 * q1; | support logic; impact depends on neighboring block and data flow. |
| 301 |         let j33 = -4.0 * q2; | support logic; impact depends on neighboring block and data flow. |
| 302 |         let j34 = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 303 | (blank) | blank separator; no runtime behavior. |
| 304 |         let mut s0 = j11 * f1 + j21 * f2 + j31 * f3 - 2.0 * bz * q2 * f4 | support logic; impact depends on neig
| 305 |             + (-2.0 * bx * q3 + 2.0 * bz * q1) * f5 | support logic; impact depends on neighboring block and da
| 306 |             + 2.0 * bx * q2 * f6; | support logic; impact depends on neighboring block and data flow. |
| 307 |         let mut s1 = j12 * f1 | support logic; impact depends on neighboring block and data flow. |
| 308 |             + j22 * f2 | support logic; impact depends on neighboring block and data flow. |
| 309 |             + j32 * f3 | support logic; impact depends on neighboring block and data flow. |
| 310 |             + 2.0 * bz * q3 * f4 | support logic; impact depends on neighboring block and data flow. |
| 311 |             + (2.0 * bx * q2 + 2.0 * bz * q0) * f5 | support logic; impact depends on neighboring block and data
| 312 |             + (2.0 * bx * q3 - 4.0 * bz * q1) * f6; | support logic; impact depends on neighboring block and da
| 313 |         let mut s2 = j13 * f1 | support logic; impact depends on neighboring block and data flow. |
| 314 |             + j23 * f2 | support logic; impact depends on neighboring block and data flow. |
```

```
| 315 |             + j33 * f3 | support logic; impact depends on neighboring block and data flow. |
| 316 |             + (-4.0 * bx * q2 - 2.0 * bz * q0) * f4 | support logic; impact depends on neighboring block and dat
| 317 |             + (2.0 * bx * q1 + 2.0 * bz * q3) * f5 | support logic; impact depends on neighboring block and data
| 318 |             + (2.0 * bx * q0 - 4.0 * bz * q2) * f6; | support logic; impact depends on neighboring block and dat
| 319 |         let mut s3 = j14 * f1 | support logic; impact depends on neighboring block and data flow. |
| 320 |             + j24 * f2 | support logic; impact depends on neighboring block and data flow. |
| 321 |             + j34 * f3 | support logic; impact depends on neighboring block and data flow. |
| 322 |             + (-4.0 * bx * q3 + 2.0 * bz * q1) * f4 | support logic; impact depends on neighboring block and dat
| 323 |             + (-2.0 * bx * q0 + 2.0 * bz * q2) * f5 | support logic; impact depends on neighboring block and dat
| 324 |             + 2.0 * bx * q1 * f6; | support logic; impact depends on neighboring block and data flow. |
| 325 |         let sn = (s0 * s0 + s1 * s1 + s2 * s2 + s3 * s3).sqrt().max(1e-10); | support logic; impact depends on r
| 326 |         s0 /= sn; | support logic; impact depends on neighboring block and data flow. |
| 327 |         s1 /= sn; | support logic; impact depends on neighboring block and data flow. |
| 328 |         s2 /= sn; | support logic; impact depends on neighboring block and data flow. |
| 329 |         s3 /= sn; | support logic; impact depends on neighboring block and data flow. |
| 330 | (blank) | blank separator; no runtime behavior. |
| 331 |         // Rate of change of quaternion from gyro | comment/documentation; changing affects understanding and in
| 332 |         let qdot0 = 0.5 * (-q1 * gx - q2 * gy - q3 * gz) - BETA * s0; | support logic; impact depends on neighbo
| 333 |         let qdot1 = 0.5 * (q0 * gx + q2 * gz - q3 * gy) - BETA * s1; | support logic; impact depends on neighbor
| 334 |         let qdot2 = 0.5 * (q0 * gy - q1 * gz + q3 * gx) - BETA * s2; | support logic; impact depends on neighbor
| 335 |         let qdot3 = 0.5 * (q0 * gz + q1 * gy - q2 * gx) - BETA * s3; | support logic; impact depends on neighbor
| 336 | (blank) | blank separator; no runtime behavior. |
| 337 |         self.q.w += qdot0 * dt; | support logic; impact depends on neighboring block and data flow. |
| 338 |         self.q.x += qdot1 * dt; | support logic; impact depends on neighboring block and data flow. |
| 339 |         self.q.y += qdot2 * dt; | support logic; impact depends on neighboring block and data flow. |
| 340 |         self.q.z += qdot3 * dt; | support logic; impact depends on neighboring block and data flow. |
| 341 |         self.q.normalise(); | support logic; impact depends on neighboring block and data flow. |
| 342 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 343 | (blank) | blank separator; no runtime behavior. |
| 344 |     fn ncyl() -> u8 { | function signature/body; changes affect API behavior and control-flow. |
| 345 |         6 | support logic; impact depends on neighboring block and data flow. |
| 346 |     } // 6-cylinder | support logic; impact depends on neighboring block and data flow. |
| 347 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/io/allowlist.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 230 | Non-empty: 205 | Symbol count: 15

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| len | 4 |
| is_allowed | 3 |
| rule_matches_frame | 3 |
| parse_u32_hex_or_dec | 3 |
| validate | 2 |
| match_hex_pattern | 2 |
| from_path | 1 |
| per_rule_rate_limit | 1 |
| parse_mask | 1 |
| can_rule_matches_mask | 1 |
| serial_rule_matches_pattern | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 10 | enum | RuleType | data/schema coupling and match/serde break risk |
| 16 | struct | AllowlistRule | data/schema coupling and match/serde break risk |
| 28 | struct | Allowlist | data/schema coupling and match/serde break risk |
| 32 | impl | Allowlist | method behavior and trait semantics |
| 33 | fn | from_path | API and behavior coupling to callers/implementers |
| 42 | fn | is_allowed | API and behavior coupling to callers/implementers |
| 46 | fn | per_rule_rate_limit | API and behavior coupling to callers/implementers |
| 53 | fn | validate | API and behavior coupling to callers/implementers |
| 95 | fn | rule_matches_frame | API and behavior coupling to callers/implementers |
| 137 | fn | parse_u32_hex_or_dec | API and behavior coupling to callers/implementers |
| 146 | fn | parse_mask | API and behavior coupling to callers/implementers |
| 150 | fn | match_hex_pattern | API and behavior coupling to callers/implementers |
| 177 | module | tests | namespace and compile-time linkage |
| 182 | fn | can_rule_matches_mask | API and behavior coupling to callers/implementers |
```

```
| 209 | fn | serial_rule_matches_pattern | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::fs; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::path::Path; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use super::hw::{Frame, HwError}; | import wiring; changing path/name can break compilation and module linkage. |
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | #[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects understan
| 9 | #[serde(rename_all = "lowercase")] | comment/documentation; changing affects understanding and intent traceability
| 10 | pub enum RuleType { | state machine variants; changes ripple through matches and business logic. |
| 11 |     Can, | support logic; impact depends on neighboring block and data flow. |
| 12 |     Serial, | support logic; impact depends on neighboring block and data flow. |
| 13 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 14 | (blank) | blank separator; no runtime behavior. |
| 15 | #[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects understa
| 16 | pub struct AllowlistRule { | data layout contract; field changes can break callers/serde/tests. |
| 17 |     #[serde(rename = "type")] | comment/documentation; changing affects understanding and intent traceability. |
| 18 |     pub rule_type: RuleType, | support logic; impact depends on neighboring block and data flow. |
| 19 |     pub id: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 20 |     pub mask: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 21 |     pub allowed_bytes: Option<Vec<[usize; 2]>>, | support logic; impact depends on neighboring block and data fl
| 22 |     pub max_rate_per_sec: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 23 |     pub pattern_hex: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 24 |     pub description: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 25 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 | #[derive(Debug, Clone, Default, PartialEq, Eq)] | comment/documentation; changing affects understanding and inter
| 28 | pub struct Allowlist { | data layout contract; field changes can break callers/serde/tests. |
| 29 |     pub rules: Vec<AllowlistRule>, | support logic; impact depends on neighboring block and data flow. |
| 30 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 | impl Allowlist { | implementation binding; behavior semantics and trait conformance live here. |
| 33 |     pub fn from_path(path: &Path) -> Result<Self, String> { | function signature/body; changes affect API behavi
| 34 |         let raw = fs::read_to_string(path).map_err(\|e\| format!("read allowlist failed: {e}"))?; | error propaga
| 35 |         let rules: Vec<AllowlistRule> = | support logic; impact depends on neighboring block and data flow. |
| 36 |             serde_json::from_str(&raw).map_err(\|e\| format!("parse allowlist failed: {e}"))?; | error propagati
| 37 |         let allow = Self { rules }; | support logic; impact depends on neighboring block and data flow. |
| 38 |         allow.validate().map_err(\|e\| e.to_string())?; | error propagation point; changes alter failure propaga
| 39 |         Ok(allow) | support logic; impact depends on neighboring block and data flow. |
| 40 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 41 | (blank) | blank separator; no runtime behavior. |
| 42 |     pub fn is_allowed(&self, frame: &Frame) -> bool { | function signature/body; changes affect API behavior and
| 43 |         self.rules.iter().any(\|r\| rule_matches_frame(r, frame)) | support logic; impact depends on neighboring
| 44 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 45 | (blank) | blank separator; no runtime behavior. |
| 46 |     pub fn per_rule_rate_limit(&self, frame: &Frame) -> Option<u32> { | function signature/body; changes affect A
| 47 |         self.rules | support logic; impact depends on neighboring block and data flow. |
| 48 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 49 |             .find(\|r\| rule_matches_frame(r, frame)) | support logic; impact depends on neighboring block and da
| 50 |             .and_then(\|r\| r.max_rate_per_sec) | support logic; impact depends on neighboring block and data fl
| 51 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 | (blank) | blank separator; no runtime behavior. |
| 53 |     fn validate(&self) -> Result<(), HwError> { | function signature/body; changes affect API behavior and contr
| 54 |         for (idx, rule) in self.rules.iter().enumerate() { | iteration logic; may alter timing, throughput, and s
| 55 |             match rule.rule_type { | branch dispatcher; arm changes alter behavior class handling. |
| 56 |                 RuleType::Can => { | support logic; impact depends on neighboring block and data flow. |
| 57 |                     if rule.id.is_none() { | runtime gate; condition changes can enable/disable critical paths.
| 58 |                         return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring bloc
| 59 |                             "allowlist rule #{idx} missing CAN id" | support logic; impact depends on neighborin
| 60 |                         ))); | support logic; impact depends on neighboring block and data flow. |
| 61 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 62 |                     if let Some(mask) = &rule.mask { | runtime gate; condition changes can enable/disable critica
| 63 |                         if parse_u32_hex_or_dec(mask).is_none() { | runtime gate; condition changes can enable/di
| 64 |                             return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring b
| 65 |                                 "allowlist rule #{idx} has invalid mask" | support logic; impact depends on neigh
| 66 |                             ))); | support logic; impact depends on neighboring block and data flow. |
| 67 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 68 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 69 |                     if let Some(windows) = &rule.allowed_bytes { | runtime gate; condition changes can enable/dis
| 70 |                         for win in windows { | iteration logic; may alter timing, throughput, and safety constra
```

```
| 71 |                            if win[1] == 0 \|\| win[0] >= 8 \|\| win[0].saturating_add(win[1]) > 8 { | runtime ga
| 72 |                                return Err(HwError::Unknown(format!( | support logic; impact depends on neighbori
| 73 |                                    "allowlist rule #{idx} has invalid allowed_bytes window" | support logic; imp
| 74 |                                ))); | support logic; impact depends on neighboring block and data flow. |
| 75 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 76 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 77 |                    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 78 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 79 |                RuleType::Serial => { | support logic; impact depends on neighboring block and data flow. |
| 80 |                    if let Some(p) = &rule.pattern_hex { | runtime gate; condition changes can enable/disable cri
| 81 |                        let compact: String = p.chars().filter(\|c\| !c.is_whitespace()).collect(); | support log
| 82 |                        if !compact.len().is_multiple_of(2) { | runtime gate; condition changes can enable/disabl
| 83 |                            return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring b
| 84 |                                "allowlist rule #{idx} has invalid serial pattern length" | support logic; impact
| 85 |                            ))); | support logic; impact depends on neighboring block and data flow. |
| 86 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 87 |                    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 88 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 |            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 90 |        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 91 |        Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 92 |    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 93 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 94 | (blank) | blank separator; no runtime behavior. |
| 95 | fn rule_matches_frame(rule: &AllowlistRule, frame: &Frame) -> bool { | function signature/body; changes affect AP
| 96 |     match (rule.rule_type.clone(), frame) { | branch dispatcher; arm changes alter behavior class handling. |
| 97 |         (RuleType::Can, Frame::Can(cf)) => { | support logic; impact depends on neighboring block and data flow.
| 98 |             let rule_id = match rule.id { | support logic; impact depends on neighboring block and data flow. |
| 99 |                 Some(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 100 |                 None => return false, | support logic; impact depends on neighboring block and data flow. |
| 101 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 102 |             let mask = rule | support logic; impact depends on neighboring block and data flow. |
| 103 |                 .mask | support logic; impact depends on neighboring block and data flow. |
| 104 |                 .as_deref() | support logic; impact depends on neighboring block and data flow. |
| 105 |                 .and_then(parse_u32_hex_or_dec) | support logic; impact depends on neighboring block and data fl
| 106 |                 .unwrap_or(0x1FFF_FFFF); | support logic; impact depends on neighboring block and data flow. |
| 107 | (blank) | blank separator; no runtime behavior. |
| 108 |             if (cf.id & mask) != (rule_id & mask) { | runtime gate; condition changes can enable/disable critica
| 109 |                 return false; | support logic; impact depends on neighboring block and data flow. |
| 110 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 111 | (blank) | blank separator; no runtime behavior. |
| 112 |             // Restrict all payload bytes outside windows when configured. | comment/documentation; changing af
| 113 |             if let Some(allowed) = &rule.allowed_bytes { | runtime gate; condition changes can enable/disable cr
| 114 |                 for i in 0..cf.len.min(8) { | iteration logic; may alter timing, throughput, and safety constrai
| 115 |                     if !allowed | runtime gate; condition changes can enable/disable critical paths. |
| 116 |                         .iter() | support logic; impact depends on neighboring block and data flow. |
| 117 |                         .any(\|w\| i >= w[0] && i < w[0].saturating_add(w[1])) | support logic; impact depends o
| 118 |                         && cf.data[i] != 0 | support logic; impact depends on neighboring block and data flow.
| 119 |                     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 120 |                         return false; | support logic; impact depends on neighboring block and data flow. |
| 121 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 122 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 123 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 124 | (blank) | blank separator; no runtime behavior. |
| 125 |             true | support logic; impact depends on neighboring block and data flow. |
| 126 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 127 |         (RuleType::Serial, Frame::Serial(sf)) => { | support logic; impact depends on neighboring block and data
| 128 |             let Some(pattern) = &rule.pattern_hex else { | support logic; impact depends on neighboring block an
| 129 |                 return false; | support logic; impact depends on neighboring block and data flow. |
| 130 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 131 |             match_hex_pattern(pattern, &sf.bytes) | support logic; impact depends on neighboring block and data
| 132 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 133 |         _ => false, | support logic; impact depends on neighboring block and data flow. |
| 134 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 135 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 136 | (blank) | blank separator; no runtime behavior. |
| 137 | fn parse_u32_hex_or_dec(s: &str) -> Option<u32> { | function signature/body; changes affect API behavior and co
| 138 |     let t = s.trim(); | support logic; impact depends on neighboring block and data flow. |
| 139 |     if let Some(hex) = t.strip_prefix("0x").or_else(\|\| t.strip_prefix("0X")) { | runtime gate; condition chang
| 140 |         u32::from_str_radix(hex, 16).ok() | support logic; impact depends on neighboring block and data flow. |
| 141 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 142 |         t.parse::<u32>().ok() | support logic; impact depends on neighboring block and data flow. |
| 143 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 144 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 145 | (blank) | blank separator; no runtime behavior. |
| 146 | pub fn parse_mask(mask: &str) -> Result<u32, HwError> { | function signature/body; changes affect API behavior a
```

```
| 147 |     parse_u32_hex_or_dec(mask).ok_or_else(\|\| HwError::Unknown("invalid allowlist mask".into())) | support logi
| 148 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 149 | (blank) | blank separator; no runtime behavior. |
| 150 | fn match_hex_pattern(pattern: &str, bytes: &[u8]) -> bool { | function signature/body; changes affect API behavi
| 151 |     let compact: String = pattern.chars().filter(\|c\| !c.is_whitespace()).collect(); | support logic; impact de
| 152 |     if !compact.len().is_multiple_of(2) { | runtime gate; condition changes can enable/disable critical paths.
| 153 |         return false; | support logic; impact depends on neighboring block and data flow. |
| 154 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 155 | (blank) | blank separator; no runtime behavior. |
| 156 |     let mut idx = 0usize; | support logic; impact depends on neighboring block and data flow. |
| 157 |     for chunk in compact.as_bytes().chunks(2) { | iteration logic; may alter timing, throughput, and safety cons
| 158 |         if idx >= bytes.len() { | runtime gate; condition changes can enable/disable critical paths. |
| 159 |             return false; | support logic; impact depends on neighboring block and data flow. |
| 160 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 161 |         let pair = std::str::from_utf8(chunk).unwrap_or_default(); | support logic; impact depends on neighborin
| 162 |         if pair != ".." { | runtime gate; condition changes can enable/disable critical paths. |
| 163 |             let Ok(expected) = u8::from_str_radix(pair, 16) else { | support logic; impact depends on neighborin
| 164 |                 return false; | support logic; impact depends on neighboring block and data flow. |
| 165 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 |             if bytes[idx] != expected { | runtime gate; condition changes can enable/disable critical paths. |
| 167 |                 return false; | support logic; impact depends on neighboring block and data flow. |
| 168 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 169 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 170 |         idx += 1; | support logic; impact depends on neighboring block and data flow. |
| 171 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 172 | (blank) | blank separator; no runtime behavior. |
| 173 |     idx == bytes.len() | support logic; impact depends on neighboring block and data flow. |
| 174 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 175 | (blank) | blank separator; no runtime behavior. |
| 176 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 177 | mod tests { | module declaration; changing can break namespace graph. |
| 178 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 179 |     use crate::io::hw::{CanFrame, Frame, SerialFrame}; | import wiring; changing path/name can break compilatio
| 180 | (blank) | blank separator; no runtime behavior. |
| 181 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 182 |     fn can_rule_matches_mask() { | function signature/body; changes affect API behavior and control-flow. |
| 183 |         let allow = Allowlist { | support logic; impact depends on neighboring block and data flow. |
| 184 |             rules: vec![AllowlistRule { | support logic; impact depends on neighboring block and data flow. |
| 185 |                 rule_type: RuleType::Can, | support logic; impact depends on neighboring block and data flow. |
| 186 |                 id: Some(0x18FF50E5), | support logic; impact depends on neighboring block and data flow. |
| 187 |                 mask: Some("0x1FFFFFFF".to_string()), | support logic; impact depends on neighboring block and d
| 188 |                 allowed_bytes: Some(vec![[2, 2]]), | support logic; impact depends on neighboring block and data
| 189 |                 max_rate_per_sec: None, | support logic; impact depends on neighboring block and data flow. |
| 190 |                 pattern_hex: None, | support logic; impact depends on neighboring block and data flow. |
| 191 |                 description: None, | support logic; impact depends on neighboring block and data flow. |
| 192 |             }], | support logic; impact depends on neighboring block and data flow. |
| 193 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 194 | (blank) | blank separator; no runtime behavior. |
| 195 |         let mut data = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 196 |         data[2] = 0xAB; | support logic; impact depends on neighboring block and data flow. |
| 197 |         let f = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 198 |             id: 0x18FF50E5, | support logic; impact depends on neighboring block and data flow. |
| 199 |             dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 200 |             data, | support logic; impact depends on neighboring block and data flow. |
| 201 |             len: 8, | support logic; impact depends on neighboring block and data flow. |
| 202 |             timestamp_ms: None, | support logic; impact depends on neighboring block and data flow. |
| 203 |         }); | support logic; impact depends on neighboring block and data flow. |
| 204 | (blank) | blank separator; no runtime behavior. |
| 205 |         assert!(allow.is_allowed(&f)); | support logic; impact depends on neighboring block and data flow. |
| 206 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 207 | (blank) | blank separator; no runtime behavior. |
| 208 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 209 |     fn serial_rule_matches_pattern() { | function signature/body; changes affect API behavior and control-flow.
| 210 |         let allow = Allowlist { | support logic; impact depends on neighboring block and data flow. |
| 211 |             rules: vec![AllowlistRule { | support logic; impact depends on neighboring block and data flow. |
| 212 |                 rule_type: RuleType::Serial, | support logic; impact depends on neighboring block and data flow
| 213 |                 id: None, | support logic; impact depends on neighboring block and data flow. |
| 214 |                 mask: None, | support logic; impact depends on neighboring block and data flow. |
| 215 |                 allowed_bytes: None, | support logic; impact depends on neighboring block and data flow. |
| 216 |                 max_rate_per_sec: None, | support logic; impact depends on neighboring block and data flow. |
| 217 |                 pattern_hex: Some("AA55..".to_string()), | support logic; impact depends on neighboring block a
| 218 |                 description: None, | support logic; impact depends on neighboring block and data flow. |
| 219 |             }], | support logic; impact depends on neighboring block and data flow. |
| 220 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 221 | (blank) | blank separator; no runtime behavior. |
| 222 |         let f = Frame::Serial(SerialFrame { | support logic; impact depends on neighboring block and data flow.
```

```
| 223 |             bytes: vec![0xAA, 0x55, 0x01], | support logic; impact depends on neighboring block and data flow.
| 224 |             protocol_hint: None, | support logic; impact depends on neighboring block and data flow. |
| 225 |             timestamp_ms: None, | support logic; impact depends on neighboring block and data flow. |
| 226 |         }); | support logic; impact depends on neighboring block and data flow. |
| 227 | (blank) | blank separator; no runtime behavior. |
| 228 |         assert!(allow.is_allowed(&f)); | support logic; impact depends on neighboring block and data flow. |
| 229 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 230 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/io/artifact.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 53 | Non-empty: 46 | Symbol count: 2

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| len | 3 |
| is_empty | 1 |
| new | 1 |
| update | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 10 | struct | FirmwareArtifactReport | data/schema coupling and match/serde break risk |
| 18 | fn | validate_firmware_artifact | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::fs; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::path::Path; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 5 | use sha2::{Digest, Sha256}; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | use super::hw::HwError; | import wiring; changing path/name can break compilation and module linkage. |
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and intent
| 10 | pub struct FirmwareArtifactReport { | data layout contract; field changes can break callers/serde/tests. |
| 11 |     pub path: String, | support logic; impact depends on neighboring block and data flow. |
| 12 |     pub bytes: usize, | support logic; impact depends on neighboring block and data flow. |
| 13 |     pub crc32: u32, | support logic; impact depends on neighboring block and data flow. |
| 14 |     pub sha256: String, | support logic; impact depends on neighboring block and data flow. |
| 15 |     pub valid: bool, | support logic; impact depends on neighboring block and data flow. |
| 16 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | pub fn validate_firmware_artifact<P: AsRef<Path>>( | function signature/body; changes affect API behavior and co
| 19 |     path: P, | support logic; impact depends on neighboring block and data flow. |
| 20 |     max_bytes: usize, | support logic; impact depends on neighboring block and data flow. |
| 21 | ) -> Result<FirmwareArtifactReport, HwError> { | support logic; impact depends on neighboring block and data flo
| 22 |     let p = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 23 |     let data = fs::read(p) | support logic; impact depends on neighboring block and data flow. |
| 24 |         .map_err(\|e\| HwError::Unknown(format!("artifact read failed ({}): {e}", p.display())))?; | error propag
| 25 | (blank) | blank separator; no runtime behavior. |
| 26 |     if data.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 27 |         return Err(HwError::Unknown("artifact payload is empty".to_string())); | support logic; impact depends o
| 28 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 29 |     if data.len() > max_bytes { | runtime gate; condition changes can enable/disable critical paths. |
| 30 |         return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring block and data flow.
| 31 |             "artifact payload too large: {} bytes (limit {})", | support logic; impact depends on neighboring bl
| 32 |             data.len(), | support logic; impact depends on neighboring block and data flow. |
| 33 |             max_bytes | support logic; impact depends on neighboring block and data flow. |
| 34 |         ))); | support logic; impact depends on neighboring block and data flow. |
| 35 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 36 | (blank) | blank separator; no runtime behavior. |
| 37 |     let crc32 = crc32fast::hash(&data); | support logic; impact depends on neighboring block and data flow. |
| 38 |     let mut hasher = Sha256::new(); | support logic; impact depends on neighboring block and data flow. |
```

```
| 39 |     hasher.update(&data); | support logic; impact depends on neighboring block and data flow. |
| 40 |     let sha = hasher.finalize(); | support logic; impact depends on neighboring block and data flow. |
| 41 |     let mut sha_hex = String::with_capacity(64); | support logic; impact depends on neighboring block and data f
| 42 |     for b in sha { | iteration logic; may alter timing, throughput, and safety constraints. |
| 43 |         sha_hex.push_str(&format!("{b:02x}")); | support logic; impact depends on neighboring block and data flo
| 44 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 45 | (blank) | blank separator; no runtime behavior. |
| 46 |     Ok(FirmwareArtifactReport { | support logic; impact depends on neighboring block and data flow. |
| 47 |         path: p.display().to_string(), | support logic; impact depends on neighboring block and data flow. |
| 48 |         bytes: data.len(), | support logic; impact depends on neighboring block and data flow. |
| 49 |         crc32, | support logic; impact depends on neighboring block and data flow. |
| 50 |         sha256: sha_hex, | support logic; impact depends on neighboring block and data flow. |
| 51 |         valid: true, | support logic; impact depends on neighboring block and data flow. |
| 52 |     }) | support logic; impact depends on neighboring block and data flow. |
| 53 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/io/ecm_params.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 157 | Non-empty: 143 | Symbol count: 11

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| decode_can_frame | 4 |
| new | 4 |
| push | 4 |
| j1939_pgn | 3 |
| merge_snapshot | 1 |
| decode_eec1_engine_speed | 1 |
| decode_et1_coolant | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 6 | const | PGN_EEC1 | namespace and compile-time linkage |
| 7 | const | PGN_ET1 | namespace and compile-time linkage |
| 8 | const | PGN_EFL_P1 | namespace and compile-time linkage |
| 11 | struct | EcmSnapshot | data/schema coupling and match/serde break risk |
| 23 | struct | DecodedParam | data/schema coupling and match/serde break risk |
| 29 | fn | decode_can_frame | API and behavior coupling to callers/implementers |
| 93 | fn | merge_snapshot | API and behavior coupling to callers/implementers |
| 111 | fn | j1939_pgn | API and behavior coupling to callers/implementers |
| 125 | module | tests | namespace and compile-time linkage |
| 129 | fn | decode_eec1_engine_speed | API and behavior coupling to callers/implementers |
| 144 | fn | decode_et1_coolant | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | use super::hw::CanFrame; | import wiring; changing path/name can break compilation and module linkage. |
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | // J1939 standard PGNs often available on heavy-duty ECM networks, including Caterpillar platforms. | comment/docu
| 6 | pub const PGN_EEC1: u32 = 61444; | support logic; impact depends on neighboring block and data flow. |
| 7 | pub const PGN_ET1: u32 = 65262; | support logic; impact depends on neighboring block and data flow. |
| 8 | pub const PGN_EFL_P1: u32 = 65263; | support logic; impact depends on neighboring block and data flow. |
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | #[derive(Debug, Clone, Serialize, Deserialize, Default, PartialEq)] | comment/documentation; changing affects un
| 11 | pub struct EcmSnapshot { | data layout contract; field changes can break callers/serde/tests. |
| 12 |     pub engine_speed_rpm: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 13 |     pub accel_pedal_pct: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 14 |     pub coolant_temp_c: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 15 |     pub oil_pressure_kpa: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub fuel_temp_c: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 17 |     pub last_seen_pgn: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 18 |     pub source_address: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 19 |     pub timestamp_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
```

```
| 20 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 21 | (blank) | blank separator; no runtime behavior. |
| 22 | #[derive(Debug, Clone, Serialize, Deserialize, PartialEq)] | comment/documentation; changing affects understandi
| 23 | pub struct DecodedParam { | data layout contract; field changes can break callers/serde/tests. |
| 24 |     pub key: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 25 |     pub value: f64, | support logic; impact depends on neighboring block and data flow. |
| 26 |     pub unit: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 27 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 28 | (blank) | blank separator; no runtime behavior. |
| 29 | pub fn decode_can_frame(frame: &CanFrame) -> Vec<DecodedParam> { | function signature/body; changes affect API be
| 30 |     let pgn = j1939_pgn(frame.id); | support logic; impact depends on neighboring block and data flow. |
| 31 |     let d = &frame.data; | support logic; impact depends on neighboring block and data flow. |
| 32 | (blank) | blank separator; no runtime behavior. |
| 33 |     match pgn { | branch dispatcher; arm changes alter behavior class handling. |
| 34 |         PGN_EEC1 => { | support logic; impact depends on neighboring block and data flow. |
| 35 |             // SPN 190 Engine Speed: bytes 4-5, little-endian, 0.125 rpm/bit. | comment/documentation; changing a
| 36 |             // SPN 91 Accelerator Pedal Position 1: byte 2, 0.4 %/bit. | comment/documentation; changing affects
| 37 |             let eng_raw = u16::from_le_bytes([d[3], d[4]]); | support logic; impact depends on neighboring block
| 38 |             let app1_raw = d[1]; | support logic; impact depends on neighboring block and data flow. |
| 39 |             let mut out = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 40 |             if eng_raw != 0xFFFF { | runtime gate; condition changes can enable/disable critical paths. |
| 41 |                 out.push(DecodedParam { | support logic; impact depends on neighboring block and data flow. |
| 42 |                     key: "engine_speed_rpm", | support logic; impact depends on neighboring block and data flow.
| 43 |                     value: f64::from(eng_raw) * 0.125, | support logic; impact depends on neighboring block and d
| 44 |                     unit: "rpm", | support logic; impact depends on neighboring block and data flow. |
| 45 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 46 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 |             if app1_raw != 0xFF { | runtime gate; condition changes can enable/disable critical paths. |
| 48 |                 out.push(DecodedParam { | support logic; impact depends on neighboring block and data flow. |
| 49 |                     key: "accel_pedal_pct", | support logic; impact depends on neighboring block and data flow.
| 50 |                     value: f64::from(app1_raw) * 0.4, | support logic; impact depends on neighboring block and da
| 51 |                     unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 52 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 53 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 |             out | support logic; impact depends on neighboring block and data flow. |
| 55 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 56 |         PGN_ET1 => { | support logic; impact depends on neighboring block and data flow. |
| 57 |             // SPN 110 Coolant Temp: byte 1, 1 C/bit, offset -40 C. | comment/documentation; changing affects und
| 58 |             // SPN 174 Fuel Temp 1: byte 2, 1 C/bit, offset -40 C. | comment/documentation; changing affects unde
| 59 |             let mut out = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 60 |             if d[0] != 0xFF { | runtime gate; condition changes can enable/disable critical paths. |
| 61 |                 out.push(DecodedParam { | support logic; impact depends on neighboring block and data flow. |
| 62 |                     key: "coolant_temp_c", | support logic; impact depends on neighboring block and data flow. |
| 63 |                     value: f64::from(d[0]) - 40.0, | support logic; impact depends on neighboring block and data
| 64 |                     unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 65 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 66 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 67 |             if d[1] != 0xFF { | runtime gate; condition changes can enable/disable critical paths. |
| 68 |                 out.push(DecodedParam { | support logic; impact depends on neighboring block and data flow. |
| 69 |                     key: "fuel_temp_c", | support logic; impact depends on neighboring block and data flow. |
| 70 |                     value: f64::from(d[1]) - 40.0, | support logic; impact depends on neighboring block and data
| 71 |                     unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 72 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 73 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 74 |             out | support logic; impact depends on neighboring block and data flow. |
| 75 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 76 |         PGN_EFL_P1 => { | support logic; impact depends on neighboring block and data flow. |
| 77 |             // SPN 100 Engine Oil Pressure: byte 4, 4 kPa/bit. | comment/documentation; changing affects understa
| 78 |             let oil_raw = d[3]; | support logic; impact depends on neighboring block and data flow. |
| 79 |             if oil_raw != 0xFF { | runtime gate; condition changes can enable/disable critical paths. |
| 80 |                 vec![DecodedParam { | support logic; impact depends on neighboring block and data flow. |
| 81 |                     key: "oil_pressure_kpa", | support logic; impact depends on neighboring block and data flow.
| 82 |                     value: f64::from(oil_raw) * 4.0, | support logic; impact depends on neighboring block and da
| 83 |                     unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 84 |                 }] | support logic; impact depends on neighboring block and data flow. |
| 85 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 86 |                 Vec::new() | support logic; impact depends on neighboring block and data flow. |
| 87 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 88 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 |         _ => Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 90 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 91 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 92 | (blank) | blank separator; no runtime behavior. |
| 93 | pub fn merge_snapshot(mut snap: EcmSnapshot, frame: &CanFrame) -> EcmSnapshot { | function signature/body; change
| 94 |     let pgn = j1939_pgn(frame.id); | support logic; impact depends on neighboring block and data flow. |
| 95 |     for p in decode_can_frame(frame) { | iteration logic; may alter timing, throughput, and safety constraints.
```

```
| 96 |         match p.key { | branch dispatcher; arm changes alter behavior class handling. |
| 97 |             "engine_speed_rpm" => snap.engine_speed_rpm = Some(p.value), | support logic; impact depends on neigh
| 98 |             "accel_pedal_pct" => snap.accel_pedal_pct = Some(p.value), | support logic; impact depends on neighbo
| 99 |             "coolant_temp_c" => snap.coolant_temp_c = Some(p.value), | support logic; impact depends on neighbori
| 100 |             "fuel_temp_c" => snap.fuel_temp_c = Some(p.value), | support logic; impact depends on neighboring b
| 101 |             "oil_pressure_kpa" => snap.oil_pressure_kpa = Some(p.value), | support logic; impact depends on neig
| 102 |             _ => {} | support logic; impact depends on neighboring block and data flow. |
| 103 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 104 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 105 |     snap.last_seen_pgn = Some(pgn); | support logic; impact depends on neighboring block and data flow. |
| 106 |     snap.source_address = Some((frame.id & 0xFF) as u8); | support logic; impact depends on neighboring block a
| 107 |     snap.timestamp_ms = frame.timestamp_ms; | support logic; impact depends on neighboring block and data flow.
| 108 |     snap | support logic; impact depends on neighboring block and data flow. |
| 109 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 110 | (blank) | blank separator; no runtime behavior. |
| 111 | pub fn j1939_pgn(can_id: u32) -> u32 { | function signature/body; changes affect API behavior and control-flow.
| 112 |     // 29-bit CAN ID for J1939: priority(3), reserved(1), dp(1), pf(8), ps(8), sa(8). | comment/documentation; c
| 113 |     let pf = ((can_id >> 16) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow.
| 114 |     let ps = ((can_id >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow.
| 115 |     let dp = (can_id >> 24) & 0x01 ; | support logic; impact depends on neighboring block and data flow. |
| 116 | (blank) | blank separator; no runtime behavior. |
| 117 |     if pf < 240 { | runtime gate; condition changes can enable/disable critical paths. |
| 118 |         (dp << 16) \| ((pf as u32) << 8) | support logic; impact depends on neighboring block and data flow. |
| 119 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 120 |         (dp << 16) \| ((pf as u32) << 8) \| ps as u32 | support logic; impact depends on neighboring block and c
| 121 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 122 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 123 | (blank) | blank separator; no runtime behavior. |
| 124 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 125 | mod tests { | module declaration; changing can break namespace graph. |
| 126 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 127 | (blank) | blank separator; no runtime behavior. |
| 128 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 129 |     fn decode_eec1_engine_speed() { | function signature/body; changes affect API behavior and control-flow. |
| 130 |         let frame = CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 131 |             id: 0x0CF00400, // PGN 61444 | support logic; impact depends on neighboring block and data flow. |
| 132 |             dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 133 |             data: [0x00, 50, 0x00, 0x40, 0x1F, 0, 0, 0], | support logic; impact depends on neighboring block a
| 134 |             len: 8, | support logic; impact depends on neighboring block and data flow. |
| 135 |             timestamp_ms: Some(1), | support logic; impact depends on neighboring block and data flow. |
| 136 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 137 | (blank) | blank separator; no runtime behavior. |
| 138 |         let vals = decode_can_frame(&frame); | support logic; impact depends on neighboring block and data flow
| 139 |         assert!(vals.iter().any(\|v\| v.key == "engine_speed_rpm")); | support logic; impact depends on neighbo
| 140 |         assert!(vals.iter().any(\|v\| v.key == "accel_pedal_pct")); | support logic; impact depends on neighbori
| 141 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 142 | (blank) | blank separator; no runtime behavior. |
| 143 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 144 |     fn decode_et1_coolant() { | function signature/body; changes affect API behavior and control-flow. |
| 145 |         let frame = CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 146 |             id: 0x18FEEE00, // PGN 65262 | support logic; impact depends on neighboring block and data flow. |
| 147 |             dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 148 |             data: [90, 100, 0, 0, 0, 0, 0, 0], | support logic; impact depends on neighboring block and data flo
| 149 |             len: 8, | support logic; impact depends on neighboring block and data flow. |
| 150 |             timestamp_ms: Some(2), | support logic; impact depends on neighboring block and data flow. |
| 151 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 152 | (blank) | blank separator; no runtime behavior. |
| 153 |         let vals = decode_can_frame(&frame); | support logic; impact depends on neighboring block and data flow
| 154 |         assert!(vals.iter().any(\|v\| v.key == "coolant_temp_c")); | support logic; impact depends on neighborin
| 155 |         assert!(vals.iter().any(\|v\| v.key == "fuel_temp_c")); | support logic; impact depends on neighboring b
| 156 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 157 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/io/hw.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 405 | Non-empty: 372 | Symbol count: 29

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 6 |
```

```
| open | 4 |
| default | 3 |
| write_effectively_enabled | 3 |
| push | 2 |
| write_intent_enabled | 2 |
| declared_capabilities | 2 |
| init | 2 |
| close | 2 |
| adapter_info | 2 |
| open_real_adapter | 2 |
| now_ms | 2 |
| missing_for_live_flash | 1 |
| from_cli_args | 1 |
| capabilities_summary | 1 |
| read_frame | 1 |
| try_read_frame | 1 |
| send_frame | 1 |
| all | 1 |
| probe_live_adapter | 1 |
| write_listen_only_startup_log | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 12 | enum | HwMode | data/schema coupling and match/serde break risk |
| 18 | struct | CanFrame | data/schema coupling and match/serde break risk |
| 27 | struct | SerialFrame | data/schema coupling and match/serde break risk |
| 34 | enum | Frame | data/schema coupling and match/serde break risk |
| 40 | struct | HwCapabilities | data/schema coupling and match/serde break risk |
| 48 | impl | HwCapabilities | method behavior and trait semantics |
| 49 | fn | missing_for_live_flash | API and behavior coupling to callers/implementers |
| 62 | struct | HwConfig | data/schema coupling and match/serde break risk |
| 99 | impl | Default for HwConfig | method behavior and trait semantics |
| 100 | fn | default | API and behavior coupling to callers/implementers |
| 140 | impl | HwConfig | method behavior and trait semantics |
| 141 | fn | from_cli_args | API and behavior coupling to callers/implementers |
| 223 | fn | write_intent_enabled | API and behavior coupling to callers/implementers |
| 227 | fn | write_effectively_enabled | API and behavior coupling to callers/implementers |
| 231 | fn | declared_capabilities | API and behavior coupling to callers/implementers |
| 253 | fn | capabilities_summary | API and behavior coupling to callers/implementers |
| 263 | enum | HwError | data/schema coupling and match/serde break risk |
| 286 | trait | HardwareInterface | API and behavior coupling to callers/implementers |
| 287 | fn | init | API and behavior coupling to callers/implementers |
| 288 | fn | read_frame | API and behavior coupling to callers/implementers |
| 289 | fn | try_read_frame | API and behavior coupling to callers/implementers |
| 290 | fn | send_frame | API and behavior coupling to callers/implementers |
| 291 | fn | close | API and behavior coupling to callers/implementers |
| 293 | fn | adapter_info | API and behavior coupling to callers/implementers |
| 298 | fn | open_real_adapter | API and behavior coupling to callers/implementers |
| 324 | fn | probe_live_adapter | API and behavior coupling to callers/implementers |
| 335 | struct | StartupAudit | data/schema coupling and match/serde break risk |
| 349 | fn | write_listen_only_startup_log | API and behavior coupling to callers/implementers |
| 400 | fn | now_ms | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::fs::{self, File, OpenOptions}; | import wiring; changing path/name can break compilation and module linka
| 2 | use std::io::Write; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | use std::path::PathBuf; | import wiring; changing path/name can break compilation and module linkage. |
| 4 | use std::time::{SystemTime, UNIX_EPOCH}; | import wiring; changing path/name can break compilation and module linl
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use flate2::write::GzEncoder; | import wiring; changing path/name can break compilation and module linkage. |
| 7 | use flate2::Compression; | import wiring; changing path/name can break compilation and module linkage. |
| 8 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 9 | use thiserror::Error; | import wiring; changing path/name can break compilation and module linkage. |
| 10 | (blank) | blank separator; no runtime behavior. |
| 11 | #[derive(Debug, Clone, Copy, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects u
| 12 | pub enum HwMode { | state machine variants; changes ripple through matches and business logic. |
| 13 |     Sim, | support logic; impact depends on neighboring block and data flow. |
| 14 |     Live, | support logic; impact depends on neighboring block and data flow. |
| 15 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 16 | (blank) | blank separator; no runtime behavior. |
```

```
| 17 | #[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects understa
| 18 | pub struct CanFrame { | data layout contract; field changes can break callers/serde/tests. |
| 19 |     pub id: u32, | support logic; impact depends on neighboring block and data flow. |
| 20 |     pub dlc: u8, | support logic; impact depends on neighboring block and data flow. |
| 21 |     pub data: [u8; 8], | support logic; impact depends on neighboring block and data flow. |
| 22 |     pub len: usize, | support logic; impact depends on neighboring block and data flow. |
| 23 |     pub timestamp_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
| 24 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 25 | (blank) | blank separator; no runtime behavior. |
| 26 | #[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects understa
| 27 | pub struct SerialFrame { | data layout contract; field changes can break callers/serde/tests. |
| 28 |     pub bytes: Vec<u8>, | support logic; impact depends on neighboring block and data flow. |
| 29 |     pub protocol_hint: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 30 |     pub timestamp_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
| 31 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 32 | (blank) | blank separator; no runtime behavior. |
| 33 | #[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects understa
| 34 | pub enum Frame { | state machine variants; changes ripple through matches and business logic. |
| 35 |     Can(CanFrame), | support logic; impact depends on neighboring block and data flow. |
| 36 |     Serial(SerialFrame), | support logic; impact depends on neighboring block and data flow. |
| 37 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | (blank) | blank separator; no runtime behavior. |
| 39 | #[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize, Default)] | comment/documentation; changing affects
| 40 | pub struct HwCapabilities { | data layout contract; field changes can break callers/serde/tests. |
| 41 |     pub can_raw: bool, | support logic; impact depends on neighboring block and data flow. |
| 42 |     pub isotp: bool, | support logic; impact depends on neighboring block and data flow. |
| 43 |     pub j1939: bool, | support logic; impact depends on neighboring block and data flow. |
| 44 |     pub uds_flash: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 45 |     pub vendor_bridge: bool, | support logic; impact depends on neighboring block and data flow. |
| 46 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 | (blank) | blank separator; no runtime behavior. |
| 48 | impl HwCapabilities { | implementation binding; behavior semantics and trait conformance live here. |
| 49 |     pub fn missing_for_live_flash(&self) -> Vec<&'static str> { | function signature/body; changes affect API bel
| 50 |         let mut missing = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 51 |         if !self.uds_flash { | runtime gate; condition changes can enable/disable critical paths. |
| 52 |             missing.push("uds_flash"); | protocol or I/O critical; changes may impact external integration corre
| 53 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 |         if self.can_raw && !(self.isotp \|\| self.vendor_bridge) { | runtime gate; condition changes can enable/
| 55 |             missing.push("isotp_or_vendor_bridge"); | support logic; impact depends on neighboring block and data
| 56 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 57 |         missing | support logic; impact depends on neighboring block and data flow. |
| 58 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 59 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 | #[derive(Debug, Clone, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects understa
| 62 | pub struct HwConfig { | data layout contract; field changes can break callers/serde/tests. |
| 63 |     pub mode: HwMode, | support logic; impact depends on neighboring block and data flow. |
| 64 |     pub vendor_name: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 65 |     pub vendor_template_dir: Option<PathBuf>, | support logic; impact depends on neighboring block and data flow
| 66 |     pub vendor_bridge_exe: Option<PathBuf>, | support logic; impact depends on neighboring block and data flow.
| 67 |     pub vendor_bridge_timeout_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 68 |     pub serial_port: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 69 |     pub serial_baud: u32, | support logic; impact depends on neighboring block and data flow. |
| 70 |     pub serial_parity: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 71 |     pub serial_stopbits: u8, | support logic; impact depends on neighboring block and data flow. |
| 72 |     pub can_interface: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 73 |     pub ethernet_probe: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 74 |     pub enable_write: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 75 |     pub allowlist_path: Option<PathBuf>, | support logic; impact depends on neighboring block and data flow. |
| 76 |     pub rate_limit_global_per_sec: u32, | support logic; impact depends on neighboring block and data flow. |
| 77 |     pub rate_limit_per_id_per_sec: u32, | support logic; impact depends on neighboring block and data flow. |
| 78 |     pub write_retry_count: u8, | protocol or I/O critical; changes may impact external integration correctness.
| 79 |     pub write_retry_backoff_ms: u64, | protocol or I/O critical; changes may impact external integration correctn
| 80 |     pub reconnect_backoff_base_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 81 |     pub reconnect_backoff_max_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 82 |     pub uds_retry_count: u8, | protocol or I/O critical; changes may impact external integration correctness. |
| 83 |     pub uds_timeout_p2_ms: u64, | protocol or I/O critical; changes may impact external integration correctness.
| 84 |     pub uds_timeout_p2star_ms: u64, | protocol or I/O critical; changes may impact external integration correctne
| 85 |     pub uds_st_min_ms: u64, | protocol or I/O critical; changes may impact external integration correctness. |
| 86 |     pub uds_block_size: u8, | protocol or I/O critical; changes may impact external integration correctness. |
| 87 |     pub j1939_idle_guard_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 88 |     pub keep_channels_alive: bool, | support logic; impact depends on neighboring block and data flow. |
| 89 |     pub parser_max_frame_size: usize, | support logic; impact depends on neighboring block and data flow. |
| 90 |     pub dry_run: bool, | support logic; impact depends on neighboring block and data flow. |
| 91 |     pub use_isolation_hw: bool, | support logic; impact depends on neighboring block and data flow. |
| 92 |     pub log_dir: PathBuf, | support logic; impact depends on neighboring block and data flow. |
```

```
| 93 |     pub metrics_enabled: bool, | support logic; impact depends on neighboring block and data flow. |
| 94 |     pub emergency_stop_hook: Option<String>, | support logic; impact depends on neighboring block and data flow.
| 95 |     pub allow_untrusted_ecu: bool, | support logic; impact depends on neighboring block and data flow. |
| 96 |     pub noninteractive_approved: bool, | support logic; impact depends on neighboring block and data flow. |
| 97 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 98 | (blank) | blank separator; no runtime behavior. |
| 99 | impl Default for HwConfig { | implementation binding; behavior semantics and trait conformance live here. |
| 100 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 101 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 102 |             mode: HwMode::Sim, | support logic; impact depends on neighboring block and data flow. |
| 103 |             vendor_name: None, | support logic; impact depends on neighboring block and data flow. |
| 104 |             vendor_template_dir: None, | support logic; impact depends on neighboring block and data flow. |
| 105 |             vendor_bridge_exe: None, | support logic; impact depends on neighboring block and data flow. |
| 106 |             vendor_bridge_timeout_ms: 1_000, | support logic; impact depends on neighboring block and data flow
| 107 |             serial_port: None, | support logic; impact depends on neighboring block and data flow. |
| 108 |             serial_baud: 115_200, | support logic; impact depends on neighboring block and data flow. |
| 109 |             serial_parity: Some("None".to_string()), | support logic; impact depends on neighboring block and da
| 110 |             serial_stopbits: 1, | support logic; impact depends on neighboring block and data flow. |
| 111 |             can_interface: None, | support logic; impact depends on neighboring block and data flow. |
| 112 |             ethernet_probe: None, | support logic; impact depends on neighboring block and data flow. |
| 113 |             enable_write: false, | protocol or I/O critical; changes may impact external integration correctness
| 114 |             allowlist_path: None, | support logic; impact depends on neighboring block and data flow. |
| 115 |             rate_limit_global_per_sec: 100, | support logic; impact depends on neighboring block and data flow.
| 116 |             rate_limit_per_id_per_sec: 10, | support logic; impact depends on neighboring block and data flow.
| 117 |             write_retry_count: 3, | protocol or I/O critical; changes may impact external integration correctnes
| 118 |             write_retry_backoff_ms: 200, | protocol or I/O critical; changes may impact external integration co
| 119 |             reconnect_backoff_base_ms: 500, | support logic; impact depends on neighboring block and data flow.
| 120 |             reconnect_backoff_max_ms: 30_000, | support logic; impact depends on neighboring block and data flow
| 121 |             uds_retry_count: 3, | protocol or I/O critical; changes may impact external integration correctness
| 122 |             uds_timeout_p2_ms: 1_000, | protocol or I/O critical; changes may impact external integration corre
| 123 |             uds_timeout_p2star_ms: 5_000, | protocol or I/O critical; changes may impact external integration co
| 124 |             uds_st_min_ms: 5, | protocol or I/O critical; changes may impact external integration correctness.
| 125 |             uds_block_size: 0, | protocol or I/O critical; changes may impact external integration correctness.
| 126 |             j1939_idle_guard_ms: 2_500, | support logic; impact depends on neighboring block and data flow. |
| 127 |             keep_channels_alive: true, | support logic; impact depends on neighboring block and data flow. |
| 128 |             parser_max_frame_size: 4096, | support logic; impact depends on neighboring block and data flow. |
| 129 |             dry_run: true, | support logic; impact depends on neighboring block and data flow. |
| 130 |             use_isolation_hw: false, | support logic; impact depends on neighboring block and data flow. |
| 131 |             log_dir: PathBuf::from("./artifacts/hw_logs"), | support logic; impact depends on neighboring block
| 132 |             metrics_enabled: false, | support logic; impact depends on neighboring block and data flow. |
| 133 |             emergency_stop_hook: None, | support logic; impact depends on neighboring block and data flow. |
| 134 |             allow_untrusted_ecu: false, | support logic; impact depends on neighboring block and data flow. |
| 135 |             noninteractive_approved: false, | support logic; impact depends on neighboring block and data flow.
| 136 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 137 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 139 | (blank) | blank separator; no runtime behavior. |
| 140 | impl HwConfig { | implementation binding; behavior semantics and trait conformance live here. |
| 141 |     pub fn from_cli_args(args: &[String]) -> Result<Self, HwError> { | function signature/body; changes affect A
| 142 |         let mut cfg = HwConfig::default(); | support logic; impact depends on neighboring block and data flow.
| 143 |         for a in args { | iteration logic; may alter timing, throughput, and safety constraints. |
| 144 |             if let Some(v) = a.strip_prefix("--hw-mode=") { | runtime gate; condition changes can enable/disable
| 145 |                 cfg.mode = match v.to_ascii_lowercase().as_str() { | support logic; impact depends on neighborin
| 146 |                     "sim" => HwMode::Sim, | support logic; impact depends on neighboring block and data flow. |
| 147 |                     "live" => HwMode::Live, | support logic; impact depends on neighboring block and data flow.
| 148 |                     _ => return Err(HwError::Unknown(format!("invalid --hw-mode value: {v}"))), | support logic
| 149 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 150 |             } else if a == "--dry-run" { | support logic; impact depends on neighboring block and data flow. |
| 151 |                 cfg.dry_run = true; | support logic; impact depends on neighboring block and data flow. |
| 152 |             } else if a == "--dry-run=false" { | support logic; impact depends on neighboring block and data flo
| 153 |                 cfg.dry_run = false; | support logic; impact depends on neighboring block and data flow. |
| 154 |             } else if a == "--enable-write" { | protocol or I/O critical; changes may impact external integratio
| 155 |                 cfg.enable_write = true; | protocol or I/O critical; changes may impact external integration co
| 156 |             } else if a == "--noninteractive-approved" { | support logic; impact depends on neighboring block a
| 157 |                 cfg.noninteractive_approved = true; | support logic; impact depends on neighboring block and da
| 158 |             } else if let Some(v) = a.strip_prefix("--allowlist=") { | support logic; impact depends on neighbo
| 159 |                 cfg.allowlist_path = Some(PathBuf::from(v)); | support logic; impact depends on neighboring bloc
| 160 |             } else if let Some(v) = a.strip_prefix("--log-dir=") { | support logic; impact depends on neighborin
| 161 |                 cfg.log_dir = PathBuf::from(v); | support logic; impact depends on neighboring block and data f
| 162 |             } else if let Some(v) = a.strip_prefix("--serial-port=") { | support logic; impact depends on neigh
| 163 |                 cfg.serial_port = Some(v.to_string()); | support logic; impact depends on neighboring block and
| 164 |             } else if let Some(v) = a.strip_prefix("--vendor-name=") { | support logic; impact depends on neigh
| 165 |                 cfg.vendor_name = Some(v.to_string()); | support logic; impact depends on neighboring block and
| 166 |             } else if let Some(v) = a.strip_prefix("--vendor-template-dir=") { | support logic; impact depends o
| 167 |                 cfg.vendor_template_dir = Some(PathBuf::from(v)); | support logic; impact depends on neighboring
| 168 |             } else if let Some(v) = a.strip_prefix("--vendor-bridge-exe=") { | support logic; impact depends on
```

```
| 169 |                 cfg.vendor_bridge_exe = Some(PathBuf::from(v)); | support logic; impact depends on neighboring b
| 170 |             } else if let Some(v) = a.strip_prefix("--vendor-bridge-timeout-ms=") { | support logic; impact depe
| 171 |                 cfg.vendor_bridge_timeout_ms = v.parse::<u64>().map_err(\|e\| { | support logic; impact depends
| 172 |                     HwError::Unknown(format!("invalid --vendor-bridge-timeout-ms value: {e}")) | support logic;
| 173 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 174 |             } else if let Some(v) = a.strip_prefix("--serial-baud=") { | support logic; impact depends on neighb
| 175 |                 cfg.serial_baud = v | support logic; impact depends on neighboring block and data flow. |
| 176 |                     .parse::<u32>() | support logic; impact depends on neighboring block and data flow. |
| 177 |                     .map_err(\|e\| HwError::Unknown(format!("invalid --serial-baud value: {e}")))?; | error prop
| 178 |             } else if let Some(v) = a.strip_prefix("--can-if=") { | support logic; impact depends on neighboring
| 179 |                 cfg.can_interface = Some(v.to_string()); | support logic; impact depends on neighboring block an
| 180 |             } else if let Some(v) = a.strip_prefix("--eth-probe=") { | support logic; impact depends on neighbo
| 181 |                 cfg.ethernet_probe = Some(v.to_string()); | support logic; impact depends on neighboring block a
| 182 |             } else if let Some(v) = a.strip_prefix("--rate-limit-global=") { | support logic; impact depends on
| 183 |                 cfg.rate_limit_global_per_sec = v.parse::<u32>().map_err(\|e\| { | support logic; impact depends
| 184 |                     HwError::Unknown(format!("invalid --rate-limit-global value: {e}")) | support logic; impact
| 185 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 186 |             } else if let Some(v) = a.strip_prefix("--rate-limit-per-id=") { | support logic; impact depends on
| 187 |                 cfg.rate_limit_per_id_per_sec = v.parse::<u32>().map_err(\|e\| { | support logic; impact depends
| 188 |                     HwError::Unknown(format!("invalid --rate-limit-per-id value: {e}")) | support logic; impact
| 189 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 190 |             } else if let Some(v) = a.strip_prefix("--uds-retries=") { | protocol or I/O critical; changes may i
| 191 |                 cfg.uds_retry_count = v | protocol or I/O critical; changes may impact external integration cor
| 192 |                     .parse::<u8>() | support logic; impact depends on neighboring block and data flow. |
| 193 |                     .map_err(\|e\| HwError::Unknown(format!("invalid --uds-retries value: {e}")))?; | error prop
| 194 |             } else if let Some(v) = a.strip_prefix("--uds-timeout-p2-ms=") { | protocol or I/O critical; changes
| 195 |                 cfg.uds_timeout_p2_ms = v.parse::<u64>().map_err(\|e\| { | protocol or I/O critical; changes may
| 196 |                     HwError::Unknown(format!("invalid --uds-timeout-p2-ms value: {e}")) | protocol or I/O critic
| 197 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 198 |             } else if let Some(v) = a.strip_prefix("--uds-timeout-p2star-ms=") { | protocol or I/O critical; cha
| 199 |                 cfg.uds_timeout_p2star_ms = v.parse::<u64>().map_err(\|e\| { | protocol or I/O critical; changes
| 200 |                     HwError::Unknown(format!("invalid --uds-timeout-p2star-ms value: {e}")) | protocol or I/O cr
| 201 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 202 |             } else if let Some(v) = a.strip_prefix("--uds-st-min-ms=") { | protocol or I/O critical; changes may
| 203 |                 cfg.uds_st_min_ms = v.parse::<u64>().map_err(\|e\| { | protocol or I/O critical; changes may imp
| 204 |                     HwError::Unknown(format!("invalid --uds-st-min-ms value: {e}")) | protocol or I/O critical;
| 205 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 206 |             } else if let Some(v) = a.strip_prefix("--uds-block-size=") { | protocol or I/O critical; changes ma
| 207 |                 cfg.uds_block_size = v.parse::<u8>().map_err(\|e\| { | protocol or I/O critical; changes may imp
| 208 |                     HwError::Unknown(format!("invalid --uds-block-size value: {e}")) | protocol or I/O critical
| 209 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 210 |             } else if let Some(v) = a.strip_prefix("--j1939-idle-guard-ms=") { | support logic; impact depends o
| 211 |                 cfg.j1939_idle_guard_ms = v.parse::<u64>().map_err(\|e\| { | support logic; impact depends on ne
| 212 |                     HwError::Unknown(format!("invalid --j1939-idle-guard-ms value: {e}")) | support logic; impac
| 213 |                 })?; | error propagation point; changes alter failure propagation semantics. |
| 214 |             } else if a == "--keep-channels-alive" { | support logic; impact depends on neighboring block and da
| 215 |                 cfg.keep_channels_alive = true; | support logic; impact depends on neighboring block and data fl
| 216 |             } else if a == "--keep-channels-alive=false" { | support logic; impact depends on neighboring block
| 217 |                 cfg.keep_channels_alive = false; | support logic; impact depends on neighboring block and data f
| 218 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 219 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 220 |         Ok(cfg) | support logic; impact depends on neighboring block and data flow. |
| 221 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 222 | (blank) | blank separator; no runtime behavior. |
| 223 |     pub fn write_intent_enabled(&self) -> bool { | function signature/body; changes affect API behavior and con
| 224 |         self.enable_write && self.allowlist_path.is_some() && self.noninteractive_approved | protocol or I/O cri
| 225 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 226 | (blank) | blank separator; no runtime behavior. |
| 227 |     pub fn write_effectively_enabled(&self) -> bool { | function signature/body; changes affect API behavior and
| 228 |         self.write_intent_enabled() && !self.dry_run | protocol or I/O critical; changes may impact external int
| 229 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 230 | (blank) | blank separator; no runtime behavior. |
| 231 |     pub fn declared_capabilities(&self) -> HwCapabilities { | function signature/body; changes affect API behavi
| 232 |         let vendor_bridge = self | support logic; impact depends on neighboring block and data flow. |
| 233 |             .vendor_name | support logic; impact depends on neighboring block and data flow. |
| 234 |             .as_deref() | support logic; impact depends on neighboring block and data flow. |
| 235 |             .map(\|v\| v.eq_ignore_ascii_case("cat_comm")) | support logic; impact depends on neighboring block
| 236 |             .unwrap_or(false) | support logic; impact depends on neighboring block and data flow. |
| 237 |             \|\| self.vendor_template_dir.is_some() | support logic; impact depends on neighboring block and dat
| 238 |             \|\| self.vendor_bridge_exe.is_some(); | support logic; impact depends on neighboring block and data
| 239 |         let can_raw = self.can_interface.is_some(); | support logic; impact depends on neighboring block and dat
| 240 |         let isotp = can_raw \|\| vendor_bridge; | support logic; impact depends on neighboring block and data fl
| 241 |         let j1939 = can_raw; | support logic; impact depends on neighboring block and data flow. |
| 242 |         let uds_flash = can_raw \|\| self.serial_port.is_some() \|\| vendor_bridge; | protocol or I/O critical;
| 243 | (blank) | blank separator; no runtime behavior. |
| 244 |         HwCapabilities { | support logic; impact depends on neighboring block and data flow. |
```

```
| 245 |             can_raw, | support logic; impact depends on neighboring block and data flow. |
| 246 |             isotp, | support logic; impact depends on neighboring block and data flow. |
| 247 |             j1939, | support logic; impact depends on neighboring block and data flow. |
| 248 |             uds_flash, | protocol or I/O critical; changes may impact external integration correctness. |
| 249 |             vendor_bridge, | support logic; impact depends on neighboring block and data flow. |
| 250 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 251 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 252 | (blank) | blank separator; no runtime behavior. |
| 253 |     pub fn capabilities_summary(&self) -> String { | function signature/body; changes affect API behavior and co
| 254 |         let c = self.declared_capabilities(); | support logic; impact depends on neighboring block and data flow
| 255 |         format!( | support logic; impact depends on neighboring block and data flow. |
| 256 |             "can_raw={} isotp={} j1939={} uds_flash={} vendor_bridge={}", | protocol or I/O critical; changes ma
| 257 |             c.can_raw, c.isotp, c.j1939, c.uds_flash, c.vendor_bridge | protocol or I/O critical; changes may im
| 258 |         ) | support logic; impact depends on neighboring block and data flow. |
| 259 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 260 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 261 | (blank) | blank separator; no runtime behavior. |
| 262 | #[derive(Debug, Error)] | comment/documentation; changing affects understanding and intent traceability. |
| 263 | pub enum HwError { | state machine variants; changes ripple through matches and business logic. |
| 264 |     #[error("port not found: {0}")] | comment/documentation; changing affects understanding and intent traceabil
| 265 |     PortNotFound(String), | support logic; impact depends on neighboring block and data flow. |
| 266 |     #[error("permission denied: {0}")] | comment/documentation; changing affects understanding and intent tracea
| 267 |     PermissionDenied(String), | support logic; impact depends on neighboring block and data flow. |
| 268 |     #[error("parse error: {cause}; raw_data={raw_data}")] | comment/documentation; changing affects understandin
| 269 |     ParseError { cause: String, raw_data: String }, | support logic; impact depends on neighboring block and dat
| 270 |     #[error("write blocked by allowlist")] | comment/documentation; changing affects understanding and intent tr
| 271 |     WriteBlockedAllowlist, | support logic; impact depends on neighboring block and data flow. |
| 272 |     #[error("rate limited")] | comment/documentation; changing affects understanding and intent traceability. |
| 273 |     RateLimited, | support logic; impact depends on neighboring block and data flow. |
| 274 |     #[error("bus off")] | comment/documentation; changing affects understanding and intent traceability. |
| 275 |     BusOff, | support logic; impact depends on neighboring block and data flow. |
| 276 |     #[error("transceiver error")] | comment/documentation; changing affects understanding and intent traceabili
| 277 |     TransceiverError, | support logic; impact depends on neighboring block and data flow. |
| 278 |     #[error("checksum error")] | comment/documentation; changing affects understanding and intent traceability.
| 279 |     ChecksumError, | support logic; impact depends on neighboring block and data flow. |
| 280 |     #[error("timeout")] | comment/documentation; changing affects understanding and intent traceability. |
| 281 |     Timeout, | support logic; impact depends on neighboring block and data flow. |
| 282 |     #[error("unknown: {0}")] | comment/documentation; changing affects understanding and intent traceability. |
| 283 |     Unknown(String), | support logic; impact depends on neighboring block and data flow. |
| 284 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 285 | (blank) | blank separator; no runtime behavior. |
| 286 | pub trait HardwareInterface { | interface contract; changes impact all implementations and users. |
| 287 |     fn init(&mut self, config: &HwConfig) -> Result<(), HwError>; | function signature/body; changes affect API
| 288 |     fn read_frame(&mut self) -> Result<Frame, HwError>; | function signature/body; changes affect API behavior a
| 289 |     fn try_read_frame(&mut self) -> Result<Option<Frame>, HwError>; | function signature/body; changes affect AP
| 290 |     fn send_frame(&mut self, frame: Frame) -> Result<(), HwError>; | function signature/body; changes affect AP
| 291 |     fn close(&mut self) -> Result<(), HwError>; | function signature/body; changes affect API behavior and contr
| 292 | (blank) | blank separator; no runtime behavior. |
| 293 |     fn adapter_info(&self) -> Option<String> { | function signature/body; changes affect API behavior and contro
| 294 |         None | support logic; impact depends on neighboring block and data flow. |
| 295 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 296 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 297 | (blank) | blank separator; no runtime behavior. |
| 298 | pub fn open_real_adapter(cfg: &HwConfig) -> Result<Box<dyn HardwareInterface>, HwError> { | function signature/b
| 299 |     #[cfg(all(target_os = "windows", feature = "vendor-windows"))] | comment/documentation; changing affects und
| 300 |     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 301 |         if let Some(vendor) = &cfg.vendor_name { | runtime gate; condition changes can enable/disable critical p
| 302 |             if vendor.eq_ignore_ascii_case("cat_comm") { | runtime gate; condition changes can enable/disable cr
| 303 |                 return super::vendor_cat_comm::CatCommAdapter::open(cfg) | support logic; impact depends on neig
| 304 |                     .map(\|ad\| Box::new(ad) as Box<dyn HardwareInterface>); | support logic; impact depends on
| 305 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 306 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 307 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 308 | (blank) | blank separator; no runtime behavior. |
| 309 |     if let Some(iface) = &cfg.can_interface { | runtime gate; condition changes can enable/disable critical path
| 310 |         let ad = super::socketcan_adapter::SocketCanAdapter::open(iface)?; | error propagation point; changes al
| 311 |         return Ok(Box::new(ad)); | support logic; impact depends on neighboring block and data flow. |
| 312 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 313 | (blank) | blank separator; no runtime behavior. |
| 314 |     if let Some(port) = &cfg.serial_port { | runtime gate; condition changes can enable/disable critical paths.
| 315 |         let ad = super::serial_adapter::SerialAdapter::open(port, cfg.serial_baud)?; | error propagation point;
| 316 |         return Ok(Box::new(ad)); | support logic; impact depends on neighboring block and data flow. |
| 317 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 318 | (blank) | blank separator; no runtime behavior. |
| 319 |     Err(HwError::PortNotFound( | support logic; impact depends on neighboring block and data flow. |
| 320 |         "no real interface configured: set --vendor-name=cat_comm with bridge/template, or --can-if, or --serial
```

```
| 321 |     )) | support logic; impact depends on neighboring block and data flow. |
| 322 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 323 | (blank) | blank separator; no runtime behavior. |
| 324 | pub fn probe_live_adapter(cfg: &HwConfig) -> Result<String, HwError> { | function signature/body; changes affec
| 325 |     let mut adapter = open_real_adapter(cfg)?; | error propagation point; changes alter failure propagation sema
| 326 |     adapter.init(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 327 |     let info = adapter | support logic; impact depends on neighboring block and data flow. |
| 328 |         .adapter_info() | support logic; impact depends on neighboring block and data flow. |
| 329 |         .unwrap_or_else(\|\| "adapter_info=unavailable".to_string()); | support logic; impact depends on neighbo
| 330 |     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 331 |     Ok(info) | support logic; impact depends on neighboring block and data flow. |
| 332 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 333 | (blank) | blank separator; no runtime behavior. |
| 334 | #[derive(Debug, Serialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 335 | struct StartupAudit<'a> { | data layout contract; field changes can break callers/serde/tests. |
| 336 |     timestamp_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 337 |     component: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 338 |     event: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 339 |     level: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 340 |     direction: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 341 |     reason: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 342 |     mode: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 343 |     dry_run: bool, | support logic; impact depends on neighboring block and data flow. |
| 344 |     enable_write: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 345 |     allowlist_present: bool, | support logic; impact depends on neighboring block and data flow. |
| 346 |     write_effectively_enabled: bool, | protocol or I/O critical; changes may impact external integration correc
| 347 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 348 | (blank) | blank separator; no runtime behavior. |
| 349 | pub fn write_listen_only_startup_log(cfg: &HwConfig) -> Result<PathBuf, HwError> { | function signature/body; ch
| 350 |     fs::create_dir_all(&cfg.log_dir) | support logic; impact depends on neighboring block and data flow. |
| 351 |         .map_err(\|e\| HwError::Unknown(format!("create log dir failed: {e}")))?; | error propagation point; cha
| 352 | (blank) | blank separator; no runtime behavior. |
| 353 |     let ts = now_ms(); | support logic; impact depends on neighboring block and data flow. |
| 354 |     let session = format!("session-{ts}"); | support logic; impact depends on neighboring block and data flow.
| 355 |     let path = cfg.log_dir.join(format!("{session}.jsonl.gz")); | support logic; impact depends on neighboring b
| 356 | (blank) | blank separator; no runtime behavior. |
| 357 |     let file = File::create(&path) | support logic; impact depends on neighboring block and data flow. |
| 358 |         .map_err(\|e\| HwError::Unknown(format!("create log file failed: {e}")))?; | error propagation point; ch
| 359 |     let mut gz = GzEncoder::new(file, Compression::default()); | support logic; impact depends on neighboring bl
| 360 | (blank) | blank separator; no runtime behavior. |
| 361 |     let mode = match cfg.mode { | support logic; impact depends on neighboring block and data flow. |
| 362 |         HwMode::Sim => "sim", | support logic; impact depends on neighboring block and data flow. |
| 363 |         HwMode::Live => "live", | support logic; impact depends on neighboring block and data flow. |
| 364 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 365 | (blank) | blank separator; no runtime behavior. |
| 366 |     let audit = StartupAudit { | support logic; impact depends on neighboring block and data flow. |
| 367 |         timestamp_ms: ts, | support logic; impact depends on neighboring block and data flow. |
| 368 |         component: "io::hw", | support logic; impact depends on neighboring block and data flow. |
| 369 |         event: "startup_policy", | support logic; impact depends on neighboring block and data flow. |
| 370 |         level: "info", | support logic; impact depends on neighboring block and data flow. |
| 371 |         direction: "blocked", | support logic; impact depends on neighboring block and data flow. |
| 372 |         reason: "safe_by_default_listen_only", | support logic; impact depends on neighboring block and data flo
| 373 |         mode, | support logic; impact depends on neighboring block and data flow. |
| 374 |         dry_run: cfg.dry_run, | support logic; impact depends on neighboring block and data flow. |
| 375 |         enable_write: cfg.enable_write, | protocol or I/O critical; changes may impact external integration cor
| 376 |         allowlist_present: cfg.allowlist_path.is_some(), | support logic; impact depends on neighboring block a
| 377 |         write_effectively_enabled: cfg.write_effectively_enabled(), | protocol or I/O critical; changes may impa
| 378 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 379 | (blank) | blank separator; no runtime behavior. |
| 380 |     let line = serde_json::to_string(&audit) | support logic; impact depends on neighboring block and data flow
| 381 |         .map_err(\|e\| HwError::Unknown(format!("serialize audit failed: {e}")))?; | error propagation point; ch
| 382 |     writeln!(gz, "{line}").map_err(\|e\| HwError::Unknown(format!("write audit failed: {e}")))?; | error propaga
| 383 |     gz.finish() | support logic; impact depends on neighboring block and data flow. |
| 384 |         .map_err(\|e\| HwError::Unknown(format!("finalize gzip failed: {e}")))?; | error propagation point; char
| 385 | (blank) | blank separator; no runtime behavior. |
| 386 |     if cfg.write_effectively_enabled() { | runtime gate; condition changes can enable/disable critical paths. |
| 387 |         let idx_path = cfg.log_dir.join("write_enabled_sessions.log"); | protocol or I/O critical; changes may i
| 388 |         let mut idx = OpenOptions::new() | support logic; impact depends on neighboring block and data flow. |
| 389 |             .create(true) | support logic; impact depends on neighboring block and data flow. |
| 390 |             .append(true) | support logic; impact depends on neighboring block and data flow. |
| 391 |             .open(idx_path) | support logic; impact depends on neighboring block and data flow. |
| 392 |             .map_err(\|e\| HwError::Unknown(format!("open write-enabled index failed: {e}")))?; | error propagat
| 393 |         writeln!(idx, "{session}") | protocol or I/O critical; changes may impact external integration correctne
| 394 |             .map_err(\|e\| HwError::Unknown(format!("append write-enabled index failed: {e}")))?; | error propag
| 395 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 396 | (blank) | blank separator; no runtime behavior. |
```

```
| 397 |     Ok(path) | support logic; impact depends on neighboring block and data flow. |
| 398 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 399 | (blank) | blank separator; no runtime behavior. |
| 400 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 401 |     SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 402 |         .duration_since(UNIX_EPOCH) | support logic; impact depends on neighboring block and data flow. |
| 403 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 404 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 405 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/io/live_runner.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 1676 | Non-empty: 1534 | Symbol count: 65

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| len | 35 |
| new | 20 |
| close | 17 |
| ensure_positive_response | 17 |
| send_uds_and_wait | 15 |
| append_flash_audit | 12 |
| now_ms | 11 |
| is_uds_response_id | 9 |
| check_live_channel_policy | 8 |
| read_frame | 8 |
| open_real_adapter | 7 |
| init | 7 |
| default | 7 |
| is_empty | 7 |
| read_did_ascii | 7 |
| optf | 6 |
| send_frame_retry | 6 |
| can_payload | 5 |
| j1939_uds_req_id | 5 |
| read_ecu_identity | 4 |
| hex_lower | 4 |
| j1939_pgn | 3 |
| append_record | 3 |
| write_effectively_enabled | 3 |
| get | 3 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 19 | const | DID_VIN | namespace and compile-time linkage |
| 20 | const | DID_SW_VERSION | namespace and compile-time linkage |
| 21 | const | DID_HW_VERSION | namespace and compile-time linkage |
| 22 | const | DID_CALIBRATION_ID | namespace and compile-time linkage |
| 23 | const | DID_ECU_SERIAL | namespace and compile-time linkage |
| 24 | const | DID_FINGERPRINT | namespace and compile-time linkage |
| 25 | const | DID_BATTERY_VOLTAGE | namespace and compile-time linkage |
| 26 | const | MIN_FLASH_SUPPLY_V | namespace and compile-time linkage |
| 31 | struct | DetectResult | data/schema coupling and match/serde break risk |
| 36 | struct | LiveFeed | data/schema coupling and match/serde break risk |
| 47 | struct | SnapshotPoint | data/schema coupling and match/serde break risk |
| 56 | impl | LiveFeed | method behavior and trait semantics |
| 57 | fn | latest_snapshot | API and behavior coupling to callers/implementers |
| 64 | fn | last_update_ms | API and behavior coupling to callers/implementers |
| 68 | fn | frames_total | API and behavior coupling to callers/implementers |
| 72 | fn | recent_points | API and behavior coupling to callers/implementers |
| 81 | fn | is_alive | API and behavior coupling to callers/implementers |
| 85 | fn | last_error | API and behavior coupling to callers/implementers |
| 92 | fn | export_csv | API and behavior coupling to callers/implementers |
| 128 | fn | stop | API and behavior coupling to callers/implementers |
| 133 | fn | detect_ecms | API and behavior coupling to callers/implementers |
| 174 | fn | connect_ecm | API and behavior coupling to callers/implementers |
| 250 | fn | start_retrieve_data | API and behavior coupling to callers/implementers |
| 373 | fn | now_ms | API and behavior coupling to callers/implementers |
```

```
| 380 | fn | optf | API and behavior coupling to callers/implementers |
| 388 | struct | FlashSummary | data/schema coupling and match/serde break risk |
| 402 | struct | FlashTransportDiagnostics | data/schema coupling and match/serde break risk |
| 417 | struct | BridgeDiagAggregate | data/schema coupling and match/serde break risk |
| 426 | struct | BridgeDiagLine | data/schema coupling and match/serde break risk |
| 436 | struct | EcuIdentity | data/schema coupling and match/serde break risk |
| 446 | struct | FlashPreflight | data/schema coupling and match/serde break risk |
| 452 | struct | FlashPreflightReport | data/schema coupling and match/serde break risk |
| 459 | enum | FlashAuditStatus | data/schema coupling and match/serde break risk |
| 465 | struct | FlashAuditEvent | data/schema coupling and match/serde break risk |
| 476 | fn | hex_lower | API and behavior coupling to callers/implementers |
| 484 | fn | append_flash_audit | API and behavior coupling to callers/implementers |
| 549 | fn | ensure_positive_response | API and behavior coupling to callers/implementers |
| 612 | fn | read_did_ascii | API and behavior coupling to callers/implementers |
| 655 | fn | read_did_battery_voltage | API and behavior coupling to callers/implementers |
| 679 | fn | decode_battery_voltage_from_can | API and behavior coupling to callers/implementers |
| 695 | fn | sample_supply_voltage_from_bus | API and behavior coupling to callers/implementers |
| 720 | fn | read_ecu_identity | API and behavior coupling to callers/implementers |
| 749 | fn | run_flash_preflight | API and behavior coupling to callers/implementers |
| 778 | fn | live_read_ecm_identity | API and behavior coupling to callers/implementers |
| 793 | fn | live_preflight_ecm | API and behavior coupling to callers/implementers |
| 816 | fn | j1939_uds_req_id | API and behavior coupling to callers/implementers |
| 820 | fn | is_uds_response_id | API and behavior coupling to callers/implementers |
| 827 | fn | can_payload | API and behavior coupling to callers/implementers |
| 831 | fn | send_frame_retry | API and behavior coupling to callers/implementers |
| 856 | fn | send_can_sf | API and behavior coupling to callers/implementers |
| 889 | fn | wait_for_fc | API and behavior coupling to callers/implementers |
| 929 | fn | send_can_multiframe | API and behavior coupling to callers/implementers |
| 1021 | fn | send_fc_cts | API and behavior coupling to callers/implementers |
| 1046 | fn | recv_can_uds_response | API and behavior coupling to callers/implementers |
| 1164 | fn | send_uds_and_wait | API and behavior coupling to callers/implementers |
| 1216 | fn | check_live_channel_policy | API and behavior coupling to callers/implementers |
| 1241 | fn | live_clean_ecm | API and behavior coupling to callers/implementers |
| 1286 | fn | live_flash_ecm_firmware | API and behavior coupling to callers/implementers |
| 1590 | fn | read_bridge_diag_since | API and behavior coupling to callers/implementers |
| 1625 | module | tests | namespace and compile-time linkage |
| 1631 | fn | uds_response_id_matches_expected_target | API and behavior coupling to callers/implementers |
| 1639 | fn | uds_response_id_rejects_wrong_pf_ps_or_sa | API and behavior coupling to callers/implementers |
| 1650 | fn | ensure_positive_response_accepts_expected_sid | API and behavior coupling to callers/implementers |
| 1658 | fn | ensure_positive_response_rejects_negative_response_code | API and behavior coupling to callers/implementer
| 1668 | fn | ensure_positive_response_rejects_unexpected_sid | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::collections::BTreeSet; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | use std::fs::{self, File}; | import wiring; changing path/name can break compilation and module linkage. |
| 4 | use std::io::Write; | import wiring; changing path/name can break compilation and module linkage. |
| 5 | use std::net::{TcpStream, ToSocketAddrs}; | import wiring; changing path/name can break compilation and module li
| 6 | use std::path::Path; | import wiring; changing path/name can break compilation and module linkage. |
| 7 | use std::sync::atomic::{AtomicBool, AtomicU64, Ordering}; | import wiring; changing path/name can break compilatio
| 8 | use std::sync::{Arc, Mutex}; | import wiring; changing path/name can break compilation and module linkage. |
| 9 | use std::sync::OnceLock; | import wiring; changing path/name can break compilation and module linkage. |
| 10 | use std::thread; | import wiring; changing path/name can break compilation and module linkage. |
| 11 | use std::time::{Duration, Instant}; | import wiring; changing path/name can break compilation and module linkage
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | use super::ecm_params::{j1939_pgn, merge_snapshot, EcmSnapshot, PGN_EEC1, PGN_EFL_P1, PGN_ET1}; | import wiring;
| 14 | use super::hw::{CanFrame, Frame, HwConfig, HwError, HwMode}; | import wiring; changing path/name can break compil
| 15 | use super::replay::{append_record, frame_to_record, JsonlRecord}; | import wiring; changing path/name can break c
| 16 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 17 | use sha2::{Digest, Sha256}; | import wiring; changing path/name can break compilation and module linkage. |
| 18 | (blank) | blank separator; no runtime behavior. |
| 19 | const DID_VIN: u16 = 0xF190; | support logic; impact depends on neighboring block and data flow. |
| 20 | const DID_SW_VERSION: u16 = 0xF189; | support logic; impact depends on neighboring block and data flow. |
| 21 | const DID_HW_VERSION: u16 = 0xF191; | support logic; impact depends on neighboring block and data flow. |
| 22 | const DID_CALIBRATION_ID: u16 = 0xF180; | support logic; impact depends on neighboring block and data flow. |
| 23 | const DID_ECU_SERIAL: u16 = 0xF18C; | support logic; impact depends on neighboring block and data flow. |
| 24 | const DID_FINGERPRINT: u16 = 0xF184; | support logic; impact depends on neighboring block and data flow. |
| 25 | const DID_BATTERY_VOLTAGE: u16 = 0xDD0E; | support logic; impact depends on neighboring block and data flow. |
| 26 | const MIN_FLASH_SUPPLY_V: f64 = 11.8; | protocol or I/O critical; changes may impact external integration correct
| 27 | (blank) | blank separator; no runtime behavior. |
| 28 | static FLASH_AUDIT_CHAIN: OnceLock<Mutex<[u8; 32]>> = OnceLock::new(); | protocol or I/O critical; changes may i
| 29 | (blank) | blank separator; no runtime behavior. |
| 30 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
```

```
| 31 | pub struct DetectResult { | data layout contract; field changes can break callers/serde/tests. |
| 32 |     pub source_addresses: Vec<u8>, | support logic; impact depends on neighboring block and data flow. |
| 33 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 | #[derive(Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 36 | pub struct LiveFeed { | data layout contract; field changes can break callers/serde/tests. |
| 37 |     stop: Arc<AtomicBool>, | support logic; impact depends on neighboring block and data flow. |
| 38 |     alive: Arc<AtomicBool>, | support logic; impact depends on neighboring block and data flow. |
| 39 |     latest_snapshot: Arc<Mutex<EcmSnapshot>>, | support logic; impact depends on neighboring block and data flow
| 40 |     history: Arc<Mutex<VecDeque<SnapshotPoint>>>, | support logic; impact depends on neighboring block and data f
| 41 |     last_error: Arc<Mutex<Option<String>>>, | support logic; impact depends on neighboring block and data flow.
| 42 |     last_update_ms: Arc<AtomicU64>, | support logic; impact depends on neighboring block and data flow. |
| 43 |     frames_total: Arc<AtomicU64>, | support logic; impact depends on neighboring block and data flow. |
| 44 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 45 | (blank) | blank separator; no runtime behavior. |
| 46 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 47 | pub struct SnapshotPoint { | data layout contract; field changes can break callers/serde/tests. |
| 48 |     pub ts_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 49 |     pub engine_speed_rpm: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 50 |     pub accel_pedal_pct: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 51 |     pub coolant_temp_c: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 52 |     pub fuel_temp_c: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 53 |     pub oil_pressure_kpa: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 54 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 | (blank) | blank separator; no runtime behavior. |
| 56 | impl LiveFeed { | implementation binding; behavior semantics and trait conformance live here. |
| 57 |     pub fn latest_snapshot(&self) -> EcmSnapshot { | function signature/body; changes affect API behavior and con
| 58 |         match self.latest_snapshot.lock() { | branch dispatcher; arm changes alter behavior class handling. |
| 59 |             Ok(s) => s.clone(), | support logic; impact depends on neighboring block and data flow. |
| 60 |             Err(poisoned) => poisoned.into_inner().clone(), | support logic; impact depends on neighboring block
| 61 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 62 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 63 | (blank) | blank separator; no runtime behavior. |
| 64 |     pub fn last_update_ms(&self) -> u64 { | function signature/body; changes affect API behavior and control-flow
| 65 |         self.last_update_ms.load(Ordering::Relaxed) | support logic; impact depends on neighboring block and data
| 66 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 67 | (blank) | blank separator; no runtime behavior. |
| 68 |     pub fn frames_total(&self) -> u64 { | function signature/body; changes affect API behavior and control-flow.
| 69 |         self.frames_total.load(Ordering::Relaxed) | support logic; impact depends on neighboring block and data f
| 70 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 | (blank) | blank separator; no runtime behavior. |
| 72 |     pub fn recent_points(&self, limit: usize) -> Vec<SnapshotPoint> { | function signature/body; changes affect A
| 73 |         let hist = match self.history.lock() { | support logic; impact depends on neighboring block and data flow
| 74 |             Ok(h) => h, | support logic; impact depends on neighboring block and data flow. |
| 75 |             Err(poisoned) => poisoned.into_inner(), | support logic; impact depends on neighboring block and data
| 76 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 77 |         let count = hist.len().min(limit); | support logic; impact depends on neighboring block and data flow. |
| 78 |         hist.iter().skip(hist.len() - count).cloned().collect() | support logic; impact depends on neighboring bl
| 79 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 80 | (blank) | blank separator; no runtime behavior. |
| 81 |     pub fn is_alive(&self) -> bool { | function signature/body; changes affect API behavior and control-flow. |
| 82 |         self.alive.load(Ordering::Relaxed) | support logic; impact depends on neighboring block and data flow. |
| 83 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 84 | (blank) | blank separator; no runtime behavior. |
| 85 |     pub fn last_error(&self) -> Option<String> { | function signature/body; changes affect API behavior and contr
| 86 |         match self.last_error.lock() { | branch dispatcher; arm changes alter behavior class handling. |
| 87 |             Ok(e) => e.clone(), | support logic; impact depends on neighboring block and data flow. |
| 88 |             Err(poisoned) => poisoned.into_inner().clone(), | support logic; impact depends on neighboring block
| 89 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 90 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 91 | (blank) | blank separator; no runtime behavior. |
| 92 |     pub fn export_csv<P: AsRef<Path>>(&self, path: P) -> Result<(), HwError> { | function signature/body; changes
| 93 |         let path = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 94 |         if let Some(parent) = path.parent() { | runtime gate; condition changes can enable/disable critical paths
| 95 |             fs::create_dir_all(parent) | support logic; impact depends on neighboring block and data flow. |
| 96 |                 .map_err(\|e\| HwError::Unknown(format!("create csv dir failed: {e}")))?; | error propagation poi
| 97 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 98 | (blank) | blank separator; no runtime behavior. |
| 99 |         let hist = self | support logic; impact depends on neighboring block and data flow. |
| 100 |             .history | support logic; impact depends on neighboring block and data flow. |
| 101 |             .lock() | support logic; impact depends on neighboring block and data flow. |
| 102 |             .map_err(\|_\| HwError::Unknown("history lock poisoned".to_string()))?; | error propagation point; c
| 103 | (blank) | blank separator; no runtime behavior. |
| 104 |         let mut f = File::create(path) | support logic; impact depends on neighboring block and data flow. |
| 105 |             .map_err(\|e\| HwError::Unknown(format!("create csv file failed: {e}")))?; | error propagation point
| 106 |         writeln!( | protocol or I/O critical; changes may impact external integration correctness. |
```

```
| 107 |             f, | support logic; impact depends on neighboring block and data flow. |
| 108 |             "ts_ms,engine_speed_rpm,accel_pedal_pct,coolant_temp_c,fuel_temp_c,oil_pressure_kpa" | support logic
| 109 |         ) | support logic; impact depends on neighboring block and data flow. |
| 110 |         .map_err(\|e\| HwError::Unknown(format!("write csv header failed: {e}")))?; | error propagation point; c
| 111 | (blank) | blank separator; no runtime behavior. |
| 112 |         for p in hist.iter() { | iteration logic; may alter timing, throughput, and safety constraints. |
| 113 |             writeln!( | protocol or I/O critical; changes may impact external integration correctness. |
| 114 |                 f, | support logic; impact depends on neighboring block and data flow. |
| 115 |                 "{},{},{},{},{},{}", | support logic; impact depends on neighboring block and data flow. |
| 116 |                 p.ts_ms, | support logic; impact depends on neighboring block and data flow. |
| 117 |                 optf(p.engine_speed_rpm), | support logic; impact depends on neighboring block and data flow. |
| 118 |                 optf(p.accel_pedal_pct), | support logic; impact depends on neighboring block and data flow. |
| 119 |                 optf(p.coolant_temp_c), | support logic; impact depends on neighboring block and data flow. |
| 120 |                 optf(p.fuel_temp_c), | support logic; impact depends on neighboring block and data flow. |
| 121 |                 optf(p.oil_pressure_kpa) | support logic; impact depends on neighboring block and data flow. |
| 122 |             ) | support logic; impact depends on neighboring block and data flow. |
| 123 |             .map_err(\|e\| HwError::Unknown(format!("write csv row failed: {e}")))?; | error propagation point;
| 124 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 125 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 126 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 127 | (blank) | blank separator; no runtime behavior. |
| 128 |     pub fn stop(&self) { | function signature/body; changes affect API behavior and control-flow. |
| 129 |         self.stop.store(true, Ordering::Relaxed); | support logic; impact depends on neighboring block and data
| 130 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 131 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 132 | (blank) | blank separator; no runtime behavior. |
| 133 | pub fn detect_ecms(cfg: &HwConfig, timeout: Duration) -> Result<DetectResult, HwError> { | function signature/b
| 134 |     if cfg.mode != HwMode::Live { | runtime gate; condition changes can enable/disable critical paths. |
| 135 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 136 |             "Detect requires --hw-mode=live".to_string(), | support logic; impact depends on neighboring block a
| 137 |         )); | support logic; impact depends on neighboring block and data flow. |
| 138 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 139 |     check_live_channel_policy(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 140 | (blank) | blank separator; no runtime behavior. |
| 141 |     let mut adapter = super::hw::open_real_adapter(cfg)?; | error propagation point; changes alter failure propa
| 142 |     adapter.init(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 143 | (blank) | blank separator; no runtime behavior. |
| 144 |     let start = Instant::now(); | support logic; impact depends on neighboring block and data flow. |
| 145 |     let mut found = BTreeSet::new(); | support logic; impact depends on neighboring block and data flow. |
| 146 | (blank) | blank separator; no runtime behavior. |
| 147 |     while start.elapsed() < timeout { | iteration logic; may alter timing, throughput, and safety constraints.
| 148 |         match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 149 |             Ok(Frame::Can(cf)) => { | support logic; impact depends on neighboring block and data flow. |
| 150 |                 let pgn = j1939_pgn(cf.id); | support logic; impact depends on neighboring block and data flow.
| 151 |                 let sa = (cf.id & 0xFF) as u8; | support logic; impact depends on neighboring block and data fl
| 152 |                 if matches!(pgn, PGN_EEC1 \| PGN_ET1 \| PGN_EFL_P1 \| 60928) { | runtime gate; condition change
| 153 |                     found.insert(sa); | support logic; impact depends on neighboring block and data flow. |
| 154 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 155 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 156 |             Ok(Frame::Serial(_)) => { | support logic; impact depends on neighboring block and data flow. |
| 157 |                 // For serial-only links, one logical ECM channel is considered detected. | comment/documentati
| 158 |                 found.insert(0); | support logic; impact depends on neighboring block and data flow. |
| 159 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 160 |             Err(HwError::Timeout) => {} | support logic; impact depends on neighboring block and data flow. |
| 161 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 162 |                 adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 163 |                 return Err(e); | support logic; impact depends on neighboring block and data flow. |
| 164 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 165 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 167 | (blank) | blank separator; no runtime behavior. |
| 168 |     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 169 |     Ok(DetectResult { | support logic; impact depends on neighboring block and data flow. |
| 170 |         source_addresses: found.into_iter().collect(), | support logic; impact depends on neighboring block and
| 171 |     }) | support logic; impact depends on neighboring block and data flow. |
| 172 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 173 | (blank) | blank separator; no runtime behavior. |
| 174 | pub fn connect_ecm(cfg: &HwConfig, target_sa: Option<u8>) -> Result<(), HwError> { | function signature/body; c
| 175 |     if cfg.mode != HwMode::Live { | runtime gate; condition changes can enable/disable critical paths. |
| 176 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 177 |             "Connect requires --hw-mode=live".to_string(), | support logic; impact depends on neighboring block
| 178 |         )); | support logic; impact depends on neighboring block and data flow. |
| 179 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 180 |     check_live_channel_policy(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 181 | (blank) | blank separator; no runtime behavior. |
| 182 |     let mut adapter = super::hw::open_real_adapter(cfg)?; | error propagation point; changes alter failure propa
```

```
| 183 |     adapter.init(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 184 | (blank) | blank separator; no runtime behavior. |
| 185 |     // For CAN links, send standard J1939 PGN requests as an integration handshake. | comment/documentation; cha
| 186 |     if cfg.can_interface.is_some() { | runtime gate; condition changes can enable/disable critical paths. |
| 187 |         let dst = target_sa.unwrap_or(0xFF); | support logic; impact depends on neighboring block and data flow
| 188 |         let reqs = [60928_u32, PGN_EEC1, PGN_ET1, PGN_EFL_P1]; | support logic; impact depends on neighboring b
| 189 |         for pgn in reqs { | iteration logic; may alter timing, throughput, and safety constraints. |
| 190 |             let mut data = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 191 |             data[0] = (pgn & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 192 |             data[1] = ((pgn >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data
| 193 |             data[2] = ((pgn >> 16) & 0xFF) as u8; | support logic; impact depends on neighboring block and data
| 194 |             let req_id = 0x18EA0000 \| ((dst as u32) << 8) \| 0xF9; // Request PGN from tool SA=0xF9 | support
| 195 |             let req = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow.
| 196 |                 id: req_id, | support logic; impact depends on neighboring block and data flow. |
| 197 |                 dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 198 |                 data, | support logic; impact depends on neighboring block and data flow. |
| 199 |                 len: 8, | support logic; impact depends on neighboring block and data flow. |
| 200 |                 timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flo
| 201 |             }); | support logic; impact depends on neighboring block and data flow. |
| 202 |             if let Err(e) = adapter.send_frame(req) { | runtime gate; condition changes can enable/disable crit
| 203 |                 adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 204 |                 return Err(e); | support logic; impact depends on neighboring block and data flow. |
| 205 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 206 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 207 | (blank) | blank separator; no runtime behavior. |
| 208 |         // Authentication/identification proxy: require at least one expected PGN | comment/documentation; chang
| 209 |         // from the requested source (or any source when no target SA is provided). | comment/documentation; cha
| 210 |         let deadline = Instant::now() + Duration::from_secs(2); | support logic; impact depends on neighboring
| 211 |         let mut got_response = false; | support logic; impact depends on neighboring block and data flow. |
| 212 |         while Instant::now() < deadline { | iteration logic; may alter timing, throughput, and safety constrain
| 213 |             match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 214 |                 Ok(Frame::Can(cf)) => { | support logic; impact depends on neighboring block and data flow. |
| 215 |                     let pgn = j1939_pgn(cf.id); | support logic; impact depends on neighboring block and data f
| 216 |                     let sa = (cf.id & 0xFF) as u8; | support logic; impact depends on neighboring block and data
| 217 |                     let valid_pgn = matches!(pgn, 60928 \| PGN_EEC1 \| PGN_ET1 \| PGN_EFL_P1); | support logic;
| 218 |                     let valid_sa = target_sa.is_none() \|\| Some(sa) == target_sa; | support logic; impact depe
| 219 |                     if valid_pgn && valid_sa { | runtime gate; condition changes can enable/disable critical pa
| 220 |                         got_response = true; | support logic; impact depends on neighboring block and data flow
| 221 |                         break; | support logic; impact depends on neighboring block and data flow. |
| 222 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 223 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 224 |                 Ok(Frame::Serial(_)) => {} | support logic; impact depends on neighboring block and data flow.
| 225 |                 Err(HwError::Timeout) => {} | support logic; impact depends on neighboring block and data flow.
| 226 |                 Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 227 |                     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 228 |                     return Err(e); | support logic; impact depends on neighboring block and data flow. |
| 229 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 230 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 231 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 232 |         if !got_response { | runtime gate; condition changes can enable/disable critical paths. |
| 233 |             adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 234 |             return Err(HwError::Timeout); | support logic; impact depends on neighboring block and data flow. |
| 235 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 236 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 237 | (blank) | blank separator; no runtime behavior. |
| 238 |     let identity = read_ecu_identity(&mut *adapter, cfg, target_sa)?; | error propagation point; changes alter
| 239 |     if !cfg.allow_untrusted_ecu && identity.fingerprint.is_none() { | runtime gate; condition changes can enabl
| 240 |         adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 241 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 242 |             "connect rejected: missing fingerprint DID 0xF184 on trusted profile".to_string(), | support logic;
| 243 |         )); | support logic; impact depends on neighboring block and data flow. |
| 244 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 245 | (blank) | blank separator; no runtime behavior. |
| 246 |     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 247 |     Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 248 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 249 | (blank) | blank separator; no runtime behavior. |
| 250 | pub fn start_retrieve_data(cfg: HwConfig, target_sa: Option<u8>) -> Result<LiveFeed, HwError> { | function signa
| 251 |     if cfg.mode != HwMode::Live { | runtime gate; condition changes can enable/disable critical paths. |
| 252 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 253 |             "Retrieve requires --hw-mode=live".to_string(), | support logic; impact depends on neighboring bloc
| 254 |         )); | support logic; impact depends on neighboring block and data flow. |
| 255 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 256 |     check_live_channel_policy(&cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 257 | (blank) | blank separator; no runtime behavior. |
| 258 |     let stop = Arc::new(AtomicBool::new(false)); | support logic; impact depends on neighboring block and data
```

```
| 259 |     let alive = Arc::new(AtomicBool::new(true)); | support logic; impact depends on neighboring block and data f
| 260 |     let latest_snapshot = Arc::new(Mutex::new(EcmSnapshot::default())); | support logic; impact depends on neigh
| 261 |     let history = Arc::new(Mutex::new(VecDeque::<SnapshotPoint>::new())); | support logic; impact depends on nei
| 262 |     let last_error = Arc::new(Mutex::new(None)); | support logic; impact depends on neighboring block and data f
| 263 |     let last_update_ms = Arc::new(AtomicU64::new(0)); | support logic; impact depends on neighboring block and d
| 264 |     let frames_total = Arc::new(AtomicU64::new(0)); | support logic; impact depends on neighboring block and dat
| 265 | (blank) | blank separator; no runtime behavior. |
| 266 |     let feed = LiveFeed { | support logic; impact depends on neighboring block and data flow. |
| 267 |         stop: Arc::clone(&stop), | support logic; impact depends on neighboring block and data flow. |
| 268 |         alive: Arc::clone(&alive), | support logic; impact depends on neighboring block and data flow. |
| 269 |         latest_snapshot: Arc::clone(&latest_snapshot), | support logic; impact depends on neighboring block and
| 270 |         history: Arc::clone(&history), | support logic; impact depends on neighboring block and data flow. |
| 271 |         last_error: Arc::clone(&last_error), | support logic; impact depends on neighboring block and data flow
| 272 |         last_update_ms: Arc::clone(&last_update_ms), | support logic; impact depends on neighboring block and da
| 273 |         frames_total: Arc::clone(&frames_total), | support logic; impact depends on neighboring block and data f
| 274 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 275 | (blank) | blank separator; no runtime behavior. |
| 276 |     thread::spawn(move \|\| { | support logic; impact depends on neighboring block and data flow. |
| 277 |         let mut adapter = match super::hw::open_real_adapter(&cfg) { | support logic; impact depends on neighbor
| 278 |             Ok(a) => a, | support logic; impact depends on neighboring block and data flow. |
| 279 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 280 |                 if let Ok(mut err) = last_error.lock() { | runtime gate; condition changes can enable/disable cr
| 281 |                     *err = Some(format!("open adapter failed: {e}")); | support logic; impact depends on neighbo
| 282 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 283 |                 alive.store(false, Ordering::Relaxed); | support logic; impact depends on neighboring block and
| 284 |                 eprintln!("[ecm-live] open adapter failed: {e}"); | support logic; impact depends on neighboring
| 285 |                 return; | support logic; impact depends on neighboring block and data flow. |
| 286 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 287 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 288 |         if let Err(e) = adapter.init(&cfg) { | runtime gate; condition changes can enable/disable critical paths
| 289 |             if let Ok(mut err) = last_error.lock() { | runtime gate; condition changes can enable/disable critic
| 290 |                 *err = Some(format!("init adapter failed: {e}")); | support logic; impact depends on neighboring
| 291 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 292 |             alive.store(false, Ordering::Relaxed); | support logic; impact depends on neighboring block and data
| 293 |             eprintln!("[ecm-live] init adapter failed: {e}"); | support logic; impact depends on neighboring bl
| 294 |             return; | support logic; impact depends on neighboring block and data flow. |
| 295 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 296 | (blank) | blank separator; no runtime behavior. |
| 297 |         let rx_log = cfg.log_dir.join("ecm_rx.jsonl"); | support logic; impact depends on neighboring block and
| 298 |         let snapshot_log = cfg.log_dir.join("ecm_snapshot.jsonl"); | support logic; impact depends on neighborin
| 299 |         let mut snapshot = EcmSnapshot::default(); | support logic; impact depends on neighboring block and data
| 300 | (blank) | blank separator; no runtime behavior. |
| 301 |         while !stop.load(Ordering::Relaxed) { | iteration logic; may alter timing, throughput, and safety constr
| 302 |             match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 303 |                 Ok(frame) => { | support logic; impact depends on neighboring block and data flow. |
| 304 |                     frames_total.fetch_add(1, Ordering::Relaxed); | support logic; impact depends on neighboring
| 305 |                     let rec = frame_to_record(&frame, "rx", None, Some(cfg.dry_run)); | support logic; impact de
| 306 |                     let _ = append_record(&rx_log, &rec); | support logic; impact depends on neighboring block a
| 307 | (blank) | blank separator; no runtime behavior. |
| 308 |                     if let Frame::Can(cf) = &frame { | runtime gate; condition changes can enable/disable critic
| 309 |                         let sa = (cf.id & 0xFF) as u8; | support logic; impact depends on neighboring block and
| 310 |                         if target_sa.is_some() && Some(sa) != target_sa { | runtime gate; condition changes can
| 311 |                             continue; | support logic; impact depends on neighboring block and data flow. |
| 312 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 313 | (blank) | blank separator; no runtime behavior. |
| 314 |                         snapshot = merge_snapshot(snapshot, cf); | support logic; impact depends on neighboring
| 315 |                         if let Ok(mut s) = latest_snapshot.lock() { | runtime gate; condition changes can enable
| 316 |                             *s = snapshot.clone(); | support logic; impact depends on neighboring block and data
| 317 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 318 | (blank) | blank separator; no runtime behavior. |
| 319 |                         if let Ok(mut hist) = history.lock() { | runtime gate; condition changes can enable/disa
| 320 |                             hist.push_back(SnapshotPoint { | support logic; impact depends on neighboring block
| 321 |                                 ts_ms: cf.timestamp_ms.unwrap_or_else(now_ms), | support logic; impact depends
| 322 |                                 engine_speed_rpm: snapshot.engine_speed_rpm, | support logic; impact depends on
| 323 |                                 accel_pedal_pct: snapshot.accel_pedal_pct, | support logic; impact depends on n
| 324 |                                 coolant_temp_c: snapshot.coolant_temp_c, | support logic; impact depends on neig
| 325 |                                 fuel_temp_c: snapshot.fuel_temp_c, | support logic; impact depends on neighborin
| 326 |                                 oil_pressure_kpa: snapshot.oil_pressure_kpa, | support logic; impact depends on
| 327 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 328 |                             if hist.len() > 5000 { | runtime gate; condition changes can enable/disable critical
| 329 |                                 let _ = hist.pop_front(); | support logic; impact depends on neighboring block a
| 330 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetime
| 331 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 332 | (blank) | blank separator; no runtime behavior. |
| 333 |                         let ts = cf.timestamp_ms.unwrap_or_else(now_ms); | support logic; impact depends on neig
| 334 |                         last_update_ms.store(ts, Ordering::Relaxed); | support logic; impact depends on neighbor
```

```
| 335 | (blank) | blank separator; no runtime behavior. |
| 336 |                         let snap_line = JsonlRecord { | support logic; impact depends on neighboring block and d
| 337 |                             ts, | support logic; impact depends on neighboring block and data flow. |
| 338 |                             transport: "can".to_string(), | support logic; impact depends on neighboring block a
| 339 |                             dir: "snapshot".to_string(), | support logic; impact depends on neighboring block an
| 340 |                             id: Some(cf.id), | support logic; impact depends on neighboring block and data flow
| 341 |                             dlc: Some(cf.dlc), | support logic; impact depends on neighboring block and data flo
| 342 |                             data: Some( | support logic; impact depends on neighboring block and data flow. |
| 343 |                                 serde_json::to_string(&snapshot) | support logic; impact depends on neighboring
| 344 |                                     .unwrap_or_else(\|_\| "{}".to_string()), | support logic; impact depends on
| 345 |                             ), | support logic; impact depends on neighboring block and data flow. |
| 346 |                             raw_hex: None, | support logic; impact depends on neighboring block and data flow.
| 347 |                             allowed: None, | support logic; impact depends on neighboring block and data flow.
| 348 |                             dry_run: Some(cfg.dry_run), | support logic; impact depends on neighboring block and
| 349 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 350 |                         let _ = append_record(&snapshot_log, &snap_line); | support logic; impact depends on ne
| 351 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 352 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 353 |                 Err(HwError::Timeout) => { | support logic; impact depends on neighboring block and data flow.
| 354 |                     thread::sleep(Duration::from_millis(20)); | support logic; impact depends on neighboring bl
| 355 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 356 |                 Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 357 |                     if let Ok(mut err) = last_error.lock() { | runtime gate; condition changes can enable/disab
| 358 |                         *err = Some(format!("read error: {e}")); | support logic; impact depends on neighboring
| 359 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 360 |                     eprintln!("[ecm-live] read error: {e}"); | support logic; impact depends on neighboring bloc
| 361 |                     thread::sleep(Duration::from_millis(150)); | support logic; impact depends on neighboring b
| 362 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 363 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 364 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 365 | (blank) | blank separator; no runtime behavior. |
| 366 |         alive.store(false, Ordering::Relaxed); | support logic; impact depends on neighboring block and data flo
| 367 |         let _ = adapter.close(); | support logic; impact depends on neighboring block and data flow. |
| 368 |     }); | support logic; impact depends on neighboring block and data flow. |
| 369 | (blank) | blank separator; no runtime behavior. |
| 370 |     Ok(feed) | support logic; impact depends on neighboring block and data flow. |
| 371 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 372 | (blank) | blank separator; no runtime behavior. |
| 373 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 374 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 375 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data flo
| 376 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 377 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 378 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 379 | (blank) | blank separator; no runtime behavior. |
| 380 | fn optf(v: Option<f64>) -> String { | function signature/body; changes affect API behavior and control-flow. |
| 381 |     match v { | branch dispatcher; arm changes alter behavior class handling. |
| 382 |         Some(x) => format!("{x:.4}"), | support logic; impact depends on neighboring block and data flow. |
| 383 |         None => String::new(), | support logic; impact depends on neighboring block and data flow. |
| 384 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 385 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 386 | (blank) | blank separator; no runtime behavior. |
| 387 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 388 | pub struct FlashSummary { | data layout contract; field changes can break callers/serde/tests. |
| 389 |     pub bytes_sent: usize, | support logic; impact depends on neighboring block and data flow. |
| 390 |     pub blocks_sent: usize, | support logic; impact depends on neighboring block and data flow. |
| 391 |     pub crc32: u32, | support logic; impact depends on neighboring block and data flow. |
| 392 |     pub supply_voltage_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 393 |     pub vin: String, | support logic; impact depends on neighboring block and data flow. |
| 394 |     pub sw_version: String, | support logic; impact depends on neighboring block and data flow. |
| 395 |     pub hw_version: String, | support logic; impact depends on neighboring block and data flow. |
| 396 |     pub calibration_id: String, | support logic; impact depends on neighboring block and data flow. |
| 397 |     pub ecu_serial: String, | support logic; impact depends on neighboring block and data flow. |
| 398 |     pub transport_diagnostics: FlashTransportDiagnostics, | protocol or I/O critical; changes may impact externa
| 399 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 400 | (blank) | blank separator; no runtime behavior. |
| 401 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 402 | pub struct FlashTransportDiagnostics { | data layout contract; field changes can break callers/serde/tests. |
| 403 |     pub security_seed_positive: bool, | support logic; impact depends on neighboring block and data flow. |
| 404 |     pub security_unlock_positive: bool, | support logic; impact depends on neighboring block and data flow. |
| 405 |     pub request_download_positive: bool, | support logic; impact depends on neighboring block and data flow. |
| 406 |     pub transfer_data_blocks_attempted: usize, | support logic; impact depends on neighboring block and data flo
| 407 |     pub transfer_data_blocks_acked: usize, | support logic; impact depends on neighboring block and data flow.
| 408 |     pub request_transfer_exit_positive: bool, | support logic; impact depends on neighboring block and data flow
| 409 |     pub fc_blocksize_seen: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 410 |     pub fc_stmin_seen_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
```

```
| 411 |     pub wait_frame_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 412 |     pub sequence_error_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 413 |     pub flowcontrol_timeout_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 414 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 415 | (blank) | blank separator; no runtime behavior. |
| 416 | #[derive(Debug, Clone, Default)] | comment/documentation; changing affects understanding and intent traceability
| 417 | struct BridgeDiagAggregate { | data layout contract; field changes can break callers/serde/tests. |
| 418 |     wait_frame_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 419 |     sequence_error_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 420 |     flowcontrol_timeout_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 421 |     fc_blocksize_seen: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 422 |     fc_stmin_seen_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
| 423 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 424 | (blank) | blank separator; no runtime behavior. |
| 425 | #[derive(Debug, Deserialize)] | comment/documentation; changing affects understanding and intent traceability.
| 426 | struct BridgeDiagLine { | data layout contract; field changes can break callers/serde/tests. |
| 427 |     ts_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 428 |     wait_frame_count_delta: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 429 |     sequence_error_count_delta: Option<u32>, | support logic; impact depends on neighboring block and data flow
| 430 |     flowcontrol_timeout_count_delta: Option<u32>, | support logic; impact depends on neighboring block and data
| 431 |     fc_blocksize: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 432 |     fc_stmin_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
| 433 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 434 | (blank) | blank separator; no runtime behavior. |
| 435 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 436 | pub struct EcuIdentity { | data layout contract; field changes can break callers/serde/tests. |
| 437 |     pub vin: String, | support logic; impact depends on neighboring block and data flow. |
| 438 |     pub sw_version: String, | support logic; impact depends on neighboring block and data flow. |
| 439 |     pub hw_version: String, | support logic; impact depends on neighboring block and data flow. |
| 440 |     pub calibration_id: String, | support logic; impact depends on neighboring block and data flow. |
| 441 |     pub ecu_serial: String, | support logic; impact depends on neighboring block and data flow. |
| 442 |     pub fingerprint: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 443 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 444 | (blank) | blank separator; no runtime behavior. |
| 445 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 446 | struct FlashPreflight { | data layout contract; field changes can break callers/serde/tests. |
| 447 |     supply_voltage_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 448 |     identity: EcuIdentity, | support logic; impact depends on neighboring block and data flow. |
| 449 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 450 | (blank) | blank separator; no runtime behavior. |
| 451 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 452 | pub struct FlashPreflightReport { | data layout contract; field changes can break callers/serde/tests. |
| 453 |     pub supply_voltage_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 454 |     pub identity: EcuIdentity, | support logic; impact depends on neighboring block and data flow. |
| 455 |     pub trusted_fingerprint: bool, | support logic; impact depends on neighboring block and data flow. |
| 456 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 457 | (blank) | blank separator; no runtime behavior. |
| 458 | #[derive(Debug, Clone, Copy, Serialize)] | comment/documentation; changing affects understanding and intent trac
| 459 | pub enum FlashAuditStatus { | state machine variants; changes ripple through matches and business logic. |
| 460 |     Ok, | support logic; impact depends on neighboring block and data flow. |
| 461 |     Failed, | support logic; impact depends on neighboring block and data flow. |
| 462 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 463 | (blank) | blank separator; no runtime behavior. |
| 464 | #[derive(Debug, Serialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 465 | struct FlashAuditEvent<'a> { | data layout contract; field changes can break callers/serde/tests. |
| 466 |     ts_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 467 |     stage: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 468 |     status: FlashAuditStatus, | protocol or I/O critical; changes may impact external integration correctness.
| 469 |     detail: String, | support logic; impact depends on neighboring block and data flow. |
| 470 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 471 |     dry_run: bool, | support logic; impact depends on neighboring block and data flow. |
| 472 |     prev_hash: String, | support logic; impact depends on neighboring block and data flow. |
| 473 |     event_hash: String, | support logic; impact depends on neighboring block and data flow. |
| 474 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 475 | (blank) | blank separator; no runtime behavior. |
| 476 | fn hex_lower(bytes: &[u8]) -> String { | function signature/body; changes affect API behavior and control-flow.
| 477 |     let mut s = String::with_capacity(bytes.len() * 2); | support logic; impact depends on neighboring block an
| 478 |     for b in bytes { | iteration logic; may alter timing, throughput, and safety constraints. |
| 479 |         s.push_str(&format!("{b:02x}")); | support logic; impact depends on neighboring block and data flow. |
| 480 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 481 |     s | support logic; impact depends on neighboring block and data flow. |
| 482 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 483 | (blank) | blank separator; no runtime behavior. |
| 484 | fn append_flash_audit( | function signature/body; changes affect API behavior and control-flow. |
| 485 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 486 |     stage: &str, | support logic; impact depends on neighboring block and data flow. |
```

```
| 487 |     status: FlashAuditStatus, | protocol or I/O critical; changes may impact external integration correctness.
| 488 |     detail: impl Into<String>, | support logic; impact depends on neighboring block and data flow. |
| 489 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 490 | ) { | support logic; impact depends on neighboring block and data flow. |
| 491 |     let chain = FLASH_AUDIT_CHAIN.get_or_init(\|\| Mutex::new([0u8; 32])); | protocol or I/O critical; changes r
| 492 |     let mut chain_guard = match chain.lock() { | support logic; impact depends on neighboring block and data flo
| 493 |         Ok(g) => g, | support logic; impact depends on neighboring block and data flow. |
| 494 |         Err(poisoned) => poisoned.into_inner(), | support logic; impact depends on neighboring block and data fl
| 495 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 496 |     let prev_hash = *chain_guard; | support logic; impact depends on neighboring block and data flow. |
| 497 |     let detail_str = detail.into(); | support logic; impact depends on neighboring block and data flow. |
| 498 |     let canonical = format!( | support logic; impact depends on neighboring block and data flow. |
| 499 |         "{}\|{}\|{:?}\|{}\|{:?}\|{}", | error propagation point; changes alter failure propagation semantics. |
| 500 |         now_ms(), | support logic; impact depends on neighboring block and data flow. |
| 501 |         stage, | support logic; impact depends on neighboring block and data flow. |
| 502 |         status, | support logic; impact depends on neighboring block and data flow. |
| 503 |         detail_str, | support logic; impact depends on neighboring block and data flow. |
| 504 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 505 |         cfg.dry_run | support logic; impact depends on neighboring block and data flow. |
| 506 |     ); | support logic; impact depends on neighboring block and data flow. |
| 507 |     let mut hasher = Sha256::new(); | support logic; impact depends on neighboring block and data flow. |
| 508 |     hasher.update(prev_hash); | support logic; impact depends on neighboring block and data flow. |
| 509 |     hasher.update(canonical.as_bytes()); | support logic; impact depends on neighboring block and data flow. |
| 510 |     let event_hash_arr: [u8; 32] = hasher.finalize().into(); | support logic; impact depends on neighboring bloc
| 511 | (blank) | blank separator; no runtime behavior. |
| 512 |     let event = FlashAuditEvent { | protocol or I/O critical; changes may impact external integration correctnes
| 513 |         ts_ms: now_ms(), | support logic; impact depends on neighboring block and data flow. |
| 514 |         stage, | support logic; impact depends on neighboring block and data flow. |
| 515 |         status, | support logic; impact depends on neighboring block and data flow. |
| 516 |         detail: detail_str, | support logic; impact depends on neighboring block and data flow. |
| 517 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 518 |         dry_run: cfg.dry_run, | support logic; impact depends on neighboring block and data flow. |
| 519 |         prev_hash: hex_lower(&prev_hash), | support logic; impact depends on neighboring block and data flow. |
| 520 |         event_hash: hex_lower(&event_hash_arr), | support logic; impact depends on neighboring block and data fl
| 521 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 522 |     let serialized = serde_json::to_string(&event).unwrap_or_else(\|_\| { | support logic; impact depends on nei
| 523 |         format!( | support logic; impact depends on neighboring block and data flow. |
| 524 |             "{{\"ts_ms\":{},\"stage\":\"{}\",\"status\":\"Failed\",\"detail\":\"audit_serialize_failed\",\"targe
| 525 |             event.ts_ms, | support logic; impact depends on neighboring block and data flow. |
| 526 |             stage, | support logic; impact depends on neighboring block and data flow. |
| 527 |             target_sa | support logic; impact depends on neighboring block and data flow. |
| 528 |                 .map(\|sa\| sa.to_string()) | support logic; impact depends on neighboring block and data flow.
| 529 |                 .unwrap_or_else(\|\| "null".to_string()), | support logic; impact depends on neighboring block a
| 530 |             cfg.dry_run | support logic; impact depends on neighboring block and data flow. |
| 531 |         ) | support logic; impact depends on neighboring block and data flow. |
| 532 |     }); | support logic; impact depends on neighboring block and data flow. |
| 533 | (blank) | blank separator; no runtime behavior. |
| 534 |     let rec = JsonlRecord { | support logic; impact depends on neighboring block and data flow. |
| 535 |         ts: now_ms(), | support logic; impact depends on neighboring block and data flow. |
| 536 |         transport: "audit".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 537 |         dir: "flash-stage".to_string(), | protocol or I/O critical; changes may impact external integration cor
| 538 |         id: None, | support logic; impact depends on neighboring block and data flow. |
| 539 |         dlc: None, | support logic; impact depends on neighboring block and data flow. |
| 540 |         data: Some(serialized), | support logic; impact depends on neighboring block and data flow. |
| 541 |         raw_hex: None, | support logic; impact depends on neighboring block and data flow. |
| 542 |         allowed: Some(cfg.write_effectively_enabled()), | protocol or I/O critical; changes may impact external
| 543 |         dry_run: Some(cfg.dry_run), | support logic; impact depends on neighboring block and data flow. |
| 544 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 545 |     let _ = append_record(&cfg.log_dir.join("ecm_flash_audit.jsonl"), &rec); | protocol or I/O critical; changes
| 546 |     *chain_guard = event_hash_arr; | support logic; impact depends on neighboring block and data flow. |
| 547 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 548 | (blank) | blank separator; no runtime behavior. |
| 549 | fn ensure_positive_response( | function signature/body; changes affect API behavior and control-flow. |
| 550 |     rsp: &[u8], | support logic; impact depends on neighboring block and data flow. |
| 551 |     req_sid: u8, | support logic; impact depends on neighboring block and data flow. |
| 552 |     stage: &str, | support logic; impact depends on neighboring block and data flow. |
| 553 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 554 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 555 | ) -> Result<(), HwError> { | support logic; impact depends on neighboring block and data flow. |
| 556 |     if rsp.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 557 |         append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 558 |             cfg, | support logic; impact depends on neighboring block and data flow. |
| 559 |             stage, | support logic; impact depends on neighboring block and data flow. |
| 560 |             FlashAuditStatus::Failed, | protocol or I/O critical; changes may impact external integration corre
| 561 |             "empty UDS response", | protocol or I/O critical; changes may impact external integration correctnes
| 562 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
```

```
| 563 |         ); | support logic; impact depends on neighboring block and data flow. |
| 564 |         return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring block and data flow
| 565 |             "{stage}: empty UDS response for service 0x{req_sid:02X}" | protocol or I/O critical; changes may im
| 566 |         ))); | support logic; impact depends on neighboring block and data flow. |
| 567 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 568 | (blank) | blank separator; no runtime behavior. |
| 569 |     if rsp[0] == 0x7F { | runtime gate; condition changes can enable/disable critical paths. |
| 570 |         let nrc = rsp.get(2).copied().unwrap_or(0x00); | support logic; impact depends on neighboring block and
| 571 |         let sid = rsp.get(1).copied().unwrap_or(0x00); | support logic; impact depends on neighboring block and
| 572 |         append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 573 |             cfg, | support logic; impact depends on neighboring block and data flow. |
| 574 |             stage, | support logic; impact depends on neighboring block and data flow. |
| 575 |             FlashAuditStatus::Failed, | protocol or I/O critical; changes may impact external integration correc
| 576 |             format!("negative response SID=0x{sid:02X} NRC=0x{nrc:02X}"), | support logic; impact depends on nei
| 577 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 578 |         ); | support logic; impact depends on neighboring block and data flow. |
| 579 |         return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring block and data flow
| 580 |             "{stage}: UDS negative response SID=0x{sid:02X}, NRC=0x{nrc:02X}" | protocol or I/O critical; change
| 581 |         ))); | support logic; impact depends on neighboring block and data flow. |
| 582 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 583 | (blank) | blank separator; no runtime behavior. |
| 584 |     let expected = req_sid.wrapping_add(0x40); | support logic; impact depends on neighboring block and data flo
| 585 |     if rsp[0] != expected { | runtime gate; condition changes can enable/disable critical paths. |
| 586 |         append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 587 |             cfg, | support logic; impact depends on neighboring block and data flow. |
| 588 |             stage, | support logic; impact depends on neighboring block and data flow. |
| 589 |             FlashAuditStatus::Failed, | protocol or I/O critical; changes may impact external integration correc
| 590 |             format!( | support logic; impact depends on neighboring block and data flow. |
| 591 |                 "unexpected positive response SID=0x{:02X}, expected=0x{expected:02X}", | support logic; impact
| 592 |                 rsp[0] | support logic; impact depends on neighboring block and data flow. |
| 593 |             ), | support logic; impact depends on neighboring block and data flow. |
| 594 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 595 |         ); | support logic; impact depends on neighboring block and data flow. |
| 596 |         return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring block and data flow
| 597 |             "{stage}: unexpected response SID=0x{:02X}, expected=0x{expected:02X}", | support logic; impact depe
| 598 |             rsp[0] | support logic; impact depends on neighboring block and data flow. |
| 599 |         ))); | support logic; impact depends on neighboring block and data flow. |
| 600 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 601 | (blank) | blank separator; no runtime behavior. |
| 602 |     append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 603 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 604 |         stage, | support logic; impact depends on neighboring block and data flow. |
| 605 |         FlashAuditStatus::Ok, | protocol or I/O critical; changes may impact external integration correctness.
| 606 |         format!("positive response SID=0x{:02X}", rsp[0]), | support logic; impact depends on neighboring block
| 607 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 608 |     ); | support logic; impact depends on neighboring block and data flow. |
| 609 |     Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 610 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 611 | (blank) | blank separator; no runtime behavior. |
| 612 | fn read_did_ascii( | function signature/body; changes affect API behavior and control-flow. |
| 613 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 614 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 615 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 616 |     did: u16, | support logic; impact depends on neighboring block and data flow. |
| 617 |     stage: &str, | support logic; impact depends on neighboring block and data flow. |
| 618 | ) -> Result<String, HwError> { | support logic; impact depends on neighboring block and data flow. |
| 619 |     let req = [0x22, (did >> 8) as u8, (did & 0xFF) as u8]; | support logic; impact depends on neighboring bloc
| 620 |     let rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration correctness
| 621 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 622 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 623 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 624 |         &req, | support logic; impact depends on neighboring block and data flow. |
| 625 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact exte
| 626 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 627 |     ensure_positive_response(&rsp, 0x22, stage, cfg, target_sa)?; | error propagation point; changes alter failu
| 628 |     if rsp.len() < 3 { | runtime gate; condition changes can enable/disable critical paths. |
| 629 |         return Err(HwError::ParseError { | support logic; impact depends on neighboring block and data flow. |
| 630 |             cause: format!("{stage}: short DID response"), | support logic; impact depends on neighboring block
| 631 |             raw_data: format!("len={}", rsp.len()), | support logic; impact depends on neighboring block and da
| 632 |         }); | support logic; impact depends on neighboring block and data flow. |
| 633 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 634 |     let rsp_did = ((rsp[1] as u16) << 8) \| rsp[2] as u16; | support logic; impact depends on neighboring block
| 635 |     if rsp_did != did { | runtime gate; condition changes can enable/disable critical paths. |
| 636 |         return Err(HwError::ParseError { | support logic; impact depends on neighboring block and data flow. |
| 637 |             cause: format!("{stage}: DID echo mismatch"), | support logic; impact depends on neighboring block a
| 638 |             raw_data: format!("expected=0x{did:04X}, got=0x{rsp_did:04X}"), | support logic; impact depends on n
```

```
| 639 |         }); | support logic; impact depends on neighboring block and data flow. |
| 640 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 641 |     let payload = &rsp[3..]; | support logic; impact depends on neighboring block and data flow. |
| 642 |     if payload.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 643 |         return Err(HwError::ParseError { | support logic; impact depends on neighboring block and data flow. |
| 644 |             cause: format!("{stage}: empty DID payload"), | support logic; impact depends on neighboring block a
| 645 |             raw_data: format!("did=0x{did:04X}"), | support logic; impact depends on neighboring block and data
| 646 |         }); | support logic; impact depends on neighboring block and data flow. |
| 647 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 648 |     let text = String::from_utf8(payload.to_vec()).map_err(\|_\| HwError::ParseError { | support logic; impact
| 649 |         cause: format!("{stage}: non-utf8 DID payload"), | support logic; impact depends on neighboring block a
| 650 |         raw_data: format!("did=0x{did:04X}, bytes={}", hex_lower(payload)), | support logic; impact depends on
| 651 |     })?; | error propagation point; changes alter failure propagation semantics. |
| 652 |     Ok(text.trim().to_string()) | support logic; impact depends on neighboring block and data flow. |
| 653 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 654 | (blank) | blank separator; no runtime behavior. |
| 655 | fn read_did_battery_voltage( | function signature/body; changes affect API behavior and control-flow. |
| 656 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 657 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 658 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 659 | ) -> Result<f64, HwError> { | support logic; impact depends on neighboring block and data flow. |
| 660 |     let req = [0x22, (DID_BATTERY_VOLTAGE >> 8) as u8, (DID_BATTERY_VOLTAGE & 0xFF) as u8]; | support logic; im
| 661 |     let rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration correctness
| 662 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 663 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 664 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 665 |         &req, | support logic; impact depends on neighboring block and data flow. |
| 666 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact exte
| 667 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 668 |     ensure_positive_response(&rsp, 0x22, "preflight::battery_did", cfg, target_sa)?; | error propagation point;
| 669 |     if rsp.len() < 5 { | runtime gate; condition changes can enable/disable critical paths. |
| 670 |         return Err(HwError::ParseError { | support logic; impact depends on neighboring block and data flow. |
| 671 |             cause: "preflight::battery_did short payload".to_string(), | support logic; impact depends on neighb
| 672 |             raw_data: format!("len={}", rsp.len()), | support logic; impact depends on neighboring block and da
| 673 |         }); | support logic; impact depends on neighboring block and data flow. |
| 674 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 675 |     let raw = u16::from_be_bytes([rsp[3], rsp[4]]); | support logic; impact depends on neighboring block and da
| 676 |     Ok((raw as f64) * 0.05) | support logic; impact depends on neighboring block and data flow. |
| 677 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 678 | (blank) | blank separator; no runtime behavior. |
| 679 | fn decode_battery_voltage_from_can(cf: &CanFrame) -> Option<f64> { | function signature/body; changes affect AP
| 680 |     let pgn = j1939_pgn(cf.id); | support logic; impact depends on neighboring block and data flow. |
| 681 |     if !matches!(pgn, 65271 \| 65272) { | runtime gate; condition changes can enable/disable critical paths. |
| 682 |         return None; | support logic; impact depends on neighboring block and data flow. |
| 683 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 684 |     let p = can_payload(cf); | support logic; impact depends on neighboring block and data flow. |
| 685 |     if p.len() < 2 { | runtime gate; condition changes can enable/disable critical paths. |
| 686 |         return None; | support logic; impact depends on neighboring block and data flow. |
| 687 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 688 |     let raw = u16::from_le_bytes([p[0], p[1]]); | support logic; impact depends on neighboring block and data f
| 689 |     if raw == 0xFFFF { | runtime gate; condition changes can enable/disable critical paths. |
| 690 |         return None; | support logic; impact depends on neighboring block and data flow. |
| 691 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 692 |     Some((raw as f64) * 0.05) | support logic; impact depends on neighboring block and data flow. |
| 693 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 694 | (blank) | blank separator; no runtime behavior. |
| 695 | fn sample_supply_voltage_from_bus( | function signature/body; changes affect API behavior and control-flow. |
| 696 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 697 |     timeout: Duration, | support logic; impact depends on neighboring block and data flow. |
| 698 | ) -> Result<Option<f64>, HwError> { | support logic; impact depends on neighboring block and data flow. |
| 699 |     let deadline = Instant::now() + timeout; | support logic; impact depends on neighboring block and data flow
| 700 |     let mut max_v = f64::NEG_INFINITY; | support logic; impact depends on neighboring block and data flow. |
| 701 |     while Instant::now() < deadline { | iteration logic; may alter timing, throughput, and safety constraints.
| 702 |         match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 703 |             Ok(Frame::Can(cf)) => { | support logic; impact depends on neighboring block and data flow. |
| 704 |                 if let Some(v) = decode_battery_voltage_from_can(&cf) { | runtime gate; condition changes can e
| 705 |                     max_v = max_v.max(v); | support logic; impact depends on neighboring block and data flow. |
| 706 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 707 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 708 |             Ok(Frame::Serial(_)) => {} | support logic; impact depends on neighboring block and data flow. |
| 709 |             Err(HwError::Timeout) => {} | support logic; impact depends on neighboring block and data flow. |
| 710 |             Err(e) => return Err(e), | support logic; impact depends on neighboring block and data flow. |
| 711 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 712 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 713 |     if max_v.is_finite() { | runtime gate; condition changes can enable/disable critical paths. |
| 714 |         Ok(Some(max_v)) | support logic; impact depends on neighboring block and data flow. |
```

```
| 715 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 716 |         Ok(None) | support logic; impact depends on neighboring block and data flow. |
| 717 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 718 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 719 | (blank) | blank separator; no runtime behavior. |
| 720 | fn read_ecu_identity( | function signature/body; changes affect API behavior and control-flow. |
| 721 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and dat
| 722 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 723 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 724 | ) -> Result<EcuIdentity, HwError> { | support logic; impact depends on neighboring block and data flow. |
| 725 |     let vin = read_did_ascii(adapter, cfg, target_sa, DID_VIN, "id::vin")?; | error propagation point; changes a
| 726 |     let sw = read_did_ascii(adapter, cfg, target_sa, DID_SW_VERSION, "id::sw")?; | error propagation point; char
| 727 |     let hw = read_did_ascii(adapter, cfg, target_sa, DID_HW_VERSION, "id::hw")?; | error propagation point; char
| 728 |     let cal = read_did_ascii(adapter, cfg, target_sa, DID_CALIBRATION_ID, "id::cal")?; | error propagation point
| 729 |     let serial = read_did_ascii(adapter, cfg, target_sa, DID_ECU_SERIAL, "id::serial")?; | error propagation poi
| 730 |     let fingerprint = read_did_ascii( | support logic; impact depends on neighboring block and data flow. |
| 731 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 732 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 733 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 734 |         DID_FINGERPRINT, | support logic; impact depends on neighboring block and data flow. |
| 735 |         "id::fingerprint", | support logic; impact depends on neighboring block and data flow. |
| 736 |     ) | support logic; impact depends on neighboring block and data flow. |
| 737 |     .ok(); | support logic; impact depends on neighboring block and data flow. |
| 738 | (blank) | blank separator; no runtime behavior. |
| 739 |     Ok(EcuIdentity { | support logic; impact depends on neighboring block and data flow. |
| 740 |         vin, | support logic; impact depends on neighboring block and data flow. |
| 741 |         sw_version: sw, | support logic; impact depends on neighboring block and data flow. |
| 742 |         hw_version: hw, | support logic; impact depends on neighboring block and data flow. |
| 743 |         calibration_id: cal, | support logic; impact depends on neighboring block and data flow. |
| 744 |         ecu_serial: serial, | support logic; impact depends on neighboring block and data flow. |
| 745 |         fingerprint, | support logic; impact depends on neighboring block and data flow. |
| 746 |     }) | support logic; impact depends on neighboring block and data flow. |
| 747 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 748 | (blank) | blank separator; no runtime behavior. |
| 749 | fn run_flash_preflight( | function signature/body; changes affect API behavior and control-flow. |
| 750 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and dat
| 751 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 752 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 753 | ) -> Result<FlashPreflight, HwError> { | protocol or I/O critical; changes may impact external integration corr
| 754 |     let supply_voltage = match sample_supply_voltage_from_bus(adapter, Duration::from_millis(500))? { | error p
| 755 |         Some(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 756 |         None => read_did_battery_voltage(adapter, cfg, target_sa)?, | error propagation point; changes alter fa
| 757 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 758 |     if supply_voltage < MIN_FLASH_SUPPLY_V { | runtime gate; condition changes can enable/disable critical paths
| 759 |         return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring block and data flow
| 760 |             "preflight failed: supply voltage {:.2}V below minimum {:.2}V", | support logic; impact depends on n
| 761 |             supply_voltage, MIN_FLASH_SUPPLY_V | protocol or I/O critical; changes may impact external integrati
| 762 |         ))); | support logic; impact depends on neighboring block and data flow. |
| 763 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 764 | (blank) | blank separator; no runtime behavior. |
| 765 |     let identity = read_ecu_identity(adapter, cfg, target_sa)?; | error propagation point; changes alter failure
| 766 |     if !cfg.allow_untrusted_ecu && identity.fingerprint.is_none() { | runtime gate; condition changes can enable
| 767 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 768 |             "preflight failed: ECU fingerprint DID 0xF184 not available; use trusted ECU profile or --allow-untr
| 769 |         )); | support logic; impact depends on neighboring block and data flow. |
| 770 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 771 | (blank) | blank separator; no runtime behavior. |
| 772 |     Ok(FlashPreflight { | protocol or I/O critical; changes may impact external integration correctness. |
| 773 |         supply_voltage_v: supply_voltage, | support logic; impact depends on neighboring block and data flow. |
| 774 |         identity, | support logic; impact depends on neighboring block and data flow. |
| 775 |     }) | support logic; impact depends on neighboring block and data flow. |
| 776 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 777 | (blank) | blank separator; no runtime behavior. |
| 778 | pub fn live_read_ecm_identity(cfg: &HwConfig, target_sa: Option<u8>) -> Result<EcuIdentity, HwError> { | functio
| 779 |     if cfg.mode != HwMode::Live { | runtime gate; condition changes can enable/disable critical paths. |
| 780 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 781 |             "identity read requires --hw-mode=live".to_string(), | support logic; impact depends on neighboring
| 782 |         )); | support logic; impact depends on neighboring block and data flow. |
| 783 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 784 |     check_live_channel_policy(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 785 | (blank) | blank separator; no runtime behavior. |
| 786 |     let mut adapter = super::hw::open_real_adapter(cfg)?; | error propagation point; changes alter failure propa
| 787 |     adapter.init(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 788 |     let identity = read_ecu_identity(&mut *adapter, cfg, target_sa)?; | error propagation point; changes alter f
| 789 |     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 790 |     Ok(identity) | support logic; impact depends on neighboring block and data flow. |
```

```
| 791 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 792 | (blank) | blank separator; no runtime behavior. |
| 793 | pub fn live_preflight_ecm( | function signature/body; changes affect API behavior and control-flow. |
| 794 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 795 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 796 | ) -> Result<FlashPreflightReport, HwError> { | protocol or I/O critical; changes may impact external integratio
| 797 |     if cfg.mode != HwMode::Live { | runtime gate; condition changes can enable/disable critical paths. |
| 798 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 799 |             "preflight requires --hw-mode=live".to_string(), | support logic; impact depends on neighboring bloc
| 800 |         )); | support logic; impact depends on neighboring block and data flow. |
| 801 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 802 |     check_live_channel_policy(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 803 | (blank) | blank separator; no runtime behavior. |
| 804 |     let mut adapter = super::hw::open_real_adapter(cfg)?; | error propagation point; changes alter failure propa
| 805 |     adapter.init(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 806 |     let preflight = run_flash_preflight(&mut *adapter, cfg, target_sa)?; | error propagation point; changes alte
| 807 |     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 808 | (blank) | blank separator; no runtime behavior. |
| 809 |     Ok(FlashPreflightReport { | protocol or I/O critical; changes may impact external integration correctness.
| 810 |         supply_voltage_v: preflight.supply_voltage_v, | support logic; impact depends on neighboring block and d
| 811 |         trusted_fingerprint: preflight.identity.fingerprint.is_some(), | support logic; impact depends on neighb
| 812 |         identity: preflight.identity, | support logic; impact depends on neighboring block and data flow. |
| 813 |     }) | support logic; impact depends on neighboring block and data flow. |
| 814 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 815 | (blank) | blank separator; no runtime behavior. |
| 816 | fn j1939_uds_req_id(dst_sa: u8) -> u32 { | function signature/body; changes affect API behavior and control-flo
| 817 |     0x18DA0000 \| ((dst_sa as u32) << 8) \| 0xF9 | support logic; impact depends on neighboring block and data f
| 818 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 819 | (blank) | blank separator; no runtime behavior. |
| 820 | fn is_uds_response_id(target_sa: Option<u8>, id: u32) -> bool { | function signature/body; changes affect API be
| 821 |     let pf = ((id >> 16) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 822 |     let ps = ((id >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 823 |     let sa = (id & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 824 |     pf == 0xDA && ps == 0xF9 && target_sa.is_none_or(\|t\| t == sa) | support logic; impact depends on neighbori
| 825 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 826 | (blank) | blank separator; no runtime behavior. |
| 827 | fn can_payload(cf: &CanFrame) -> &[u8] { | function signature/body; changes affect API behavior and control-flo
| 828 |     &cf.data[..cf.len.min(8)] | support logic; impact depends on neighboring block and data flow. |
| 829 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 830 | (blank) | blank separator; no runtime behavior. |
| 831 | fn send_frame_retry( | function signature/body; changes affect API behavior and control-flow. |
| 832 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 833 |     frame: Frame, | support logic; impact depends on neighboring block and data flow. |
| 834 |     retry_count: u8, | support logic; impact depends on neighboring block and data flow. |
| 835 |     backoff_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 836 | ) -> Result<(), HwError> { | support logic; impact depends on neighboring block and data flow. |
| 837 |     let max_tries = retry_count.saturating_add(1); | support logic; impact depends on neighboring block and data
| 838 |     let mut try_idx = 0u8; | support logic; impact depends on neighboring block and data flow. |
| 839 |     loop { | iteration logic; may alter timing, throughput, and safety constraints. |
| 840 |         match adapter.send_frame(frame.clone()) { | branch dispatcher; arm changes alter behavior class handling
| 841 |             Ok(()) => return Ok(()), | support logic; impact depends on neighboring block and data flow. |
| 842 |             Err(HwError::Timeout) | support logic; impact depends on neighboring block and data flow. |
| 843 |             \| Err(HwError::RateLimited) | support logic; impact depends on neighboring block and data flow. |
| 844 |             \| Err(HwError::TransceiverError) | support logic; impact depends on neighboring block and data flo
| 845 |             \| Err(HwError::BusOff) | support logic; impact depends on neighboring block and data flow. |
| 846 |                 if try_idx + 1 < max_tries => | runtime gate; condition changes can enable/disable critical pat
| 847 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 848 |                 try_idx = try_idx.saturating_add(1); | support logic; impact depends on neighboring block and d
| 849 |                 thread::sleep(Duration::from_millis(backoff_ms.max(1))); | support logic; impact depends on nei
| 850 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 851 |             Err(e) => return Err(e), | support logic; impact depends on neighboring block and data flow. |
| 852 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 853 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 854 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 855 | (blank) | blank separator; no runtime behavior. |
| 856 | fn send_can_sf( | function signature/body; changes affect API behavior and control-flow. |
| 857 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 858 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 859 |     dst_sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 860 |     payload: &[u8], | support logic; impact depends on neighboring block and data flow. |
| 861 | ) -> Result<(), HwError> { | support logic; impact depends on neighboring block and data flow. |
| 862 |     if payload.len() > 7 { | runtime gate; condition changes can enable/disable critical paths. |
| 863 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 864 |             "CAN single-frame payload exceeds 7 bytes".to_string(), | support logic; impact depends on neighbori
| 865 |         )); | support logic; impact depends on neighboring block and data flow. |
| 866 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 867 | (blank) | blank separator; no runtime behavior. |
| 868 |     let mut data = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 869 |     data[0] = payload.len() as u8; | support logic; impact depends on neighboring block and data flow. |
| 870 |     if !payload.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 871 |         data[1..(1 + payload.len())].copy_from_slice(payload); | support logic; impact depends on neighboring bl
| 872 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 873 | (blank) | blank separator; no runtime behavior. |
| 874 |     let frame = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 875 |         id: j1939_uds_req_id(dst_sa), | protocol or I/O critical; changes may impact external integration corre
| 876 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 877 |         data, | support logic; impact depends on neighboring block and data flow. |
| 878 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 879 |         timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 880 |     }); | support logic; impact depends on neighboring block and data flow. |
| 881 |     send_frame_retry( | protocol or I/O critical; changes may impact external integration correctness. |
| 882 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 883 |         frame, | support logic; impact depends on neighboring block and data flow. |
| 884 |         cfg.uds_retry_count, | protocol or I/O critical; changes may impact external integration correctness. |
| 885 |         cfg.write_retry_backoff_ms, | protocol or I/O critical; changes may impact external integration correctn
| 886 |     ) | support logic; impact depends on neighboring block and data flow. |
| 887 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 888 | (blank) | blank separator; no runtime behavior. |
| 889 | fn wait_for_fc( | function signature/body; changes affect API behavior and control-flow. |
| 890 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 891 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 892 |     target_sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 893 |     timeout: Duration, | support logic; impact depends on neighboring block and data flow. |
| 894 | ) -> Result<(u8, u64), HwError> { | support logic; impact depends on neighboring block and data flow. |
| 895 |     let deadline = Instant::now() + timeout; | support logic; impact depends on neighboring block and data flow
| 896 |     while Instant::now() < deadline { | iteration logic; may alter timing, throughput, and safety constraints.
| 897 |         match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 898 |             Ok(Frame::Can(cf)) => { | support logic; impact depends on neighboring block and data flow. |
| 899 |                 if !is_uds_response_id(Some(target_sa), cf.id) { | runtime gate; condition changes can enable/di
| 900 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 901 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 902 |                 let p = can_payload(&cf); | support logic; impact depends on neighboring block and data flow. |
| 903 |                 if p.len() < 3 { | runtime gate; condition changes can enable/disable critical paths. |
| 904 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 905 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 906 |                 let pci_type = (p[0] >> 4) & 0x0F; | support logic; impact depends on neighboring block and data
| 907 |                 if pci_type != 0x3 { | runtime gate; condition changes can enable/disable critical paths. |
| 908 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 909 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 910 |                 let fs = p[0] & 0x0F; | support logic; impact depends on neighboring block and data flow. |
| 911 |                 if fs == 0x1 { | runtime gate; condition changes can enable/disable critical paths. |
| 912 |                     return Err(HwError::RateLimited); | support logic; impact depends on neighboring block and
| 913 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 914 |                 if fs == 0x2 { | runtime gate; condition changes can enable/disable critical paths. |
| 915 |                     return Err(HwError::Timeout); | support logic; impact depends on neighboring block and data
| 916 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 917 |                 let bs = p[1]; | support logic; impact depends on neighboring block and data flow. |
| 918 |                 let st_min = p[2] as u64; | support logic; impact depends on neighboring block and data flow. |
| 919 |                 return Ok((bs, st_min.max(cfg.uds_st_min_ms))); | protocol or I/O critical; changes may impact
| 920 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 921 |             Ok(_) => {} | support logic; impact depends on neighboring block and data flow. |
| 922 |             Err(HwError::Timeout) => {} | support logic; impact depends on neighboring block and data flow. |
| 923 |             Err(e) => return Err(e), | support logic; impact depends on neighboring block and data flow. |
| 924 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 925 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 926 |     Err(HwError::Timeout) | support logic; impact depends on neighboring block and data flow. |
| 927 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 928 | (blank) | blank separator; no runtime behavior. |
| 929 | fn send_can_multiframe( | function signature/body; changes affect API behavior and control-flow. |
| 930 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 931 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 932 |     dst_sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 933 |     payload: &[u8], | support logic; impact depends on neighboring block and data flow. |
| 934 | ) -> Result<(), HwError> { | support logic; impact depends on neighboring block and data flow. |
| 935 |     if payload.len() <= 7 { | runtime gate; condition changes can enable/disable critical paths. |
| 936 |         return send_can_sf(adapter, cfg, dst_sa, payload); | protocol or I/O critical; changes may impact exter
| 937 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 938 |     if payload.len() > 4095 { | runtime gate; condition changes can enable/disable critical paths. |
| 939 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 940 |             "CAN ISO-TP payload too large (>4095 bytes)".to_string(), | support logic; impact depends on neighb
| 941 |         )); | support logic; impact depends on neighboring block and data flow. |
| 942 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 943 | (blank) | blank separator; no runtime behavior. |
| 944 |     let total_len = payload.len(); | support logic; impact depends on neighboring block and data flow. |
| 945 |     let mut ff = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 946 |     ff[0] = 0x10 \| (((total_len >> 8) & 0x0F) as u8); | support logic; impact depends on neighboring block and
| 947 |     ff[1] = (total_len & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 948 |     ff[2..8].copy_from_slice(&payload[..6]); | support logic; impact depends on neighboring block and data flow
| 949 | (blank) | blank separator; no runtime behavior. |
| 950 |     let ff_frame = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 951 |         id: j1939_uds_req_id(dst_sa), | protocol or I/O critical; changes may impact external integration corre
| 952 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 953 |         data: ff, | support logic; impact depends on neighboring block and data flow. |
| 954 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 955 |         timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 956 |     }); | support logic; impact depends on neighboring block and data flow. |
| 957 |     send_frame_retry( | protocol or I/O critical; changes may impact external integration correctness. |
| 958 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 959 |         ff_frame, | support logic; impact depends on neighboring block and data flow. |
| 960 |         cfg.uds_retry_count, | protocol or I/O critical; changes may impact external integration correctness. |
| 961 |         cfg.write_retry_backoff_ms, | protocol or I/O critical; changes may impact external integration correctn
| 962 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 963 | (blank) | blank separator; no runtime behavior. |
| 964 |     let (mut bs, mut st_min) = wait_for_fc( | support logic; impact depends on neighboring block and data flow.
| 965 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 966 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 967 |         dst_sa, | support logic; impact depends on neighboring block and data flow. |
| 968 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact exte
| 969 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 970 |     if cfg.uds_block_size > 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 971 |         bs = cfg.uds_block_size; | protocol or I/O critical; changes may impact external integration correctness
| 972 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 973 |     st_min = st_min.max(cfg.uds_st_min_ms); | protocol or I/O critical; changes may impact external integration
| 974 | (blank) | blank separator; no runtime behavior. |
| 975 |     let mut idx = 6usize; | support logic; impact depends on neighboring block and data flow. |
| 976 |     let mut sn = 1u8; | support logic; impact depends on neighboring block and data flow. |
| 977 |     let mut sent_in_block = 0u8; | support logic; impact depends on neighboring block and data flow. |
| 978 | (blank) | blank separator; no runtime behavior. |
| 979 |     while idx < payload.len() { | iteration logic; may alter timing, throughput, and safety constraints. |
| 980 |         let end = (idx + 7).min(payload.len()); | support logic; impact depends on neighboring block and data f
| 981 |         let mut data = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 982 |         data[0] = 0x20 \| (sn & 0x0F); | support logic; impact depends on neighboring block and data flow. |
| 983 |         let count = end - idx; | support logic; impact depends on neighboring block and data flow. |
| 984 |         data[1..(1 + count)].copy_from_slice(&payload[idx..end]); | support logic; impact depends on neighborin
| 985 | (blank) | blank separator; no runtime behavior. |
| 986 |         let cf = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 987 |             id: j1939_uds_req_id(dst_sa), | protocol or I/O critical; changes may impact external integration c
| 988 |             dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 989 |             data, | support logic; impact depends on neighboring block and data flow. |
| 990 |             len: 8, | support logic; impact depends on neighboring block and data flow. |
| 991 |             timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 992 |         }); | support logic; impact depends on neighboring block and data flow. |
| 993 |         send_frame_retry(adapter, cf, cfg.uds_retry_count, cfg.write_retry_backoff_ms)?; | error propagation poi
| 994 | (blank) | blank separator; no runtime behavior. |
| 995 |         idx = end; | support logic; impact depends on neighboring block and data flow. |
| 996 |         sn = (sn + 1) & 0x0F; | support logic; impact depends on neighboring block and data flow. |
| 997 |         sent_in_block = sent_in_block.saturating_add(1); | support logic; impact depends on neighboring block a
| 998 | (blank) | blank separator; no runtime behavior. |
| 999 |         if st_min > 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 1000 |             thread::sleep(Duration::from_millis(st_min)); | support logic; impact depends on neighboring block
| 1001 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1002 | (blank) | blank separator; no runtime behavior. |
| 1003 |         if bs > 0 && sent_in_block >= bs && idx < payload.len() { | runtime gate; condition changes can enable
| 1004 |             let (new_bs, new_st) = wait_for_fc( | support logic; impact depends on neighboring block and data
| 1005 |                 adapter, | support logic; impact depends on neighboring block and data flow. |
| 1006 |                 cfg, | support logic; impact depends on neighboring block and data flow. |
| 1007 |                 dst_sa, | support logic; impact depends on neighboring block and data flow. |
| 1008 |                 Duration::from_millis(cfg.uds_timeout_p2star_ms.max(1)), | protocol or I/O critical; changes m
| 1009 |             )?; | error propagation point; changes alter failure propagation semantics. |
| 1010 |             sent_in_block = 0; | support logic; impact depends on neighboring block and data flow. |
| 1011 |             if cfg.uds_block_size == 0 { | runtime gate; condition changes can enable/disable critical paths.
| 1012 |                 bs = new_bs; | support logic; impact depends on neighboring block and data flow. |
| 1013 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1014 |             st_min = new_st.max(cfg.uds_st_min_ms); | protocol or I/O critical; changes may impact external in
| 1015 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1016 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1017 | (blank) | blank separator; no runtime behavior. |
| 1018 |     Ok(()) | support logic; impact depends on neighboring block and data flow. |
```

```
| 1019 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1020 | (blank) | blank separator; no runtime behavior. |
| 1021 | fn send_fc_cts( | function signature/body; changes affect API behavior and control-flow. |
| 1022 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 1023 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 1024 |     target_sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 1025 | ) -> Result<(), HwError> { | support logic; impact depends on neighboring block and data flow. |
| 1026 |     let mut data = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1027 |     data[0] = 0x30; | support logic; impact depends on neighboring block and data flow. |
| 1028 |     data[1] = cfg.uds_block_size; | protocol or I/O critical; changes may impact external integration correctne
| 1029 |     data[2] = cfg.uds_st_min_ms.min(0x7F) as u8; | protocol or I/O critical; changes may impact external integ
| 1030 | (blank) | blank separator; no runtime behavior. |
| 1031 |     let frame = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 1032 |         id: j1939_uds_req_id(target_sa), | protocol or I/O critical; changes may impact external integration co
| 1033 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 1034 |         data, | support logic; impact depends on neighboring block and data flow. |
| 1035 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 1036 |         timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 1037 |     }); | support logic; impact depends on neighboring block and data flow. |
| 1038 |     send_frame_retry( | protocol or I/O critical; changes may impact external integration correctness. |
| 1039 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 1040 |         frame, | support logic; impact depends on neighboring block and data flow. |
| 1041 |         cfg.uds_retry_count, | protocol or I/O critical; changes may impact external integration correctness.
| 1042 |         cfg.write_retry_backoff_ms, | protocol or I/O critical; changes may impact external integration correct
| 1043 |     ) | support logic; impact depends on neighboring block and data flow. |
| 1044 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1045 | (blank) | blank separator; no runtime behavior. |
| 1046 | fn recv_can_uds_response( | function signature/body; changes affect API behavior and control-flow. |
| 1047 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 1048 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 1049 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 1050 |     timeout: Duration, | support logic; impact depends on neighboring block and data flow. |
| 1051 | ) -> Result<Vec<u8>, HwError> { | support logic; impact depends on neighboring block and data flow. |
| 1052 |     let mut deadline = Instant::now() + timeout; | support logic; impact depends on neighboring block and data
| 1053 |     let mut last_keepalive = Instant::now(); | support logic; impact depends on neighboring block and data flo
| 1054 | (blank) | blank separator; no runtime behavior. |
| 1055 |     loop { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1056 |         if Instant::now() >= deadline { | runtime gate; condition changes can enable/disable critical paths. |
| 1057 |             return Err(HwError::Timeout); | support logic; impact depends on neighboring block and data flow.
| 1058 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1059 | (blank) | blank separator; no runtime behavior. |
| 1060 |         match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 1061 |             Ok(Frame::Can(cf)) => { | support logic; impact depends on neighboring block and data flow. |
| 1062 |                 if !is_uds_response_id(target_sa, cf.id) { | runtime gate; condition changes can enable/disable
| 1063 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 1064 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1065 |                 let p = can_payload(&cf); | support logic; impact depends on neighboring block and data flow.
| 1066 |                 if p.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 1067 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 1068 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1069 | (blank) | blank separator; no runtime behavior. |
| 1070 |                 let pci_type = (p[0] >> 4) & 0x0F; | support logic; impact depends on neighboring block and da
| 1071 |                 if pci_type == 0x0 { | runtime gate; condition changes can enable/disable critical paths. |
| 1072 |                     let l = (p[0] & 0x0F) as usize; | support logic; impact depends on neighboring block and da
| 1073 |                     if p.len() < 1 + l { | runtime gate; condition changes can enable/disable critical paths.
| 1074 |                         continue; | support logic; impact depends on neighboring block and data flow. |
| 1075 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1076 |                     let payload = p[1..(1 + l)].to_vec(); | support logic; impact depends on neighboring block
| 1077 |                     if payload.len() >= 3 && payload[0] == 0x7F && payload[2] == 0x78 { | runtime gate; condit
| 1078 |                         deadline = Instant::now() | support logic; impact depends on neighboring block and data
| 1079 |                             + Duration::from_millis(cfg.uds_timeout_p2star_ms.max(1)); | protocol or I/O critic
| 1080 |                         continue; | support logic; impact depends on neighboring block and data flow. |
| 1081 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1082 |                     return Ok(payload); | support logic; impact depends on neighboring block and data flow. |
| 1083 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1084 | (blank) | blank separator; no runtime behavior. |
| 1085 |                 if pci_type == 0x1 { | runtime gate; condition changes can enable/disable critical paths. |
| 1086 |                     if p.len() < 2 { | runtime gate; condition changes can enable/disable critical paths. |
| 1087 |                         continue; | support logic; impact depends on neighboring block and data flow. |
| 1088 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1089 |                     let total_len = (((p[0] as usize) & 0x0F) << 8) \| (p[1] as usize); | support logic; impact
| 1090 |                     if total_len == 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 1091 |                         continue; | support logic; impact depends on neighboring block and data flow. |
| 1092 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1093 | (blank) | blank separator; no runtime behavior. |
| 1094 |                     let src_sa = (cf.id & 0xFF) as u8; | support logic; impact depends on neighboring block an
```

```
| 1095 |                     send_fc_cts(adapter, cfg, src_sa)?; | error propagation point; changes alter failure propag
| 1096 | (blank) | blank separator; no runtime behavior. |
| 1097 |                     let mut out = Vec::with_capacity(total_len); | support logic; impact depends on neighboring
| 1098 |                     out.extend_from_slice(&p[2..p.len().min(8)]); | support logic; impact depends on neighborin
| 1099 |                     let mut expected_sn = 1u8; | support logic; impact depends on neighboring block and data fl
| 1100 | (blank) | blank separator; no runtime behavior. |
| 1101 |                     while out.len() < total_len { | iteration logic; may alter timing, throughput, and safety c
| 1102 |                         match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class hand
| 1103 |                             Ok(Frame::Can(seg)) => { | support logic; impact depends on neighboring block and c
| 1104 |                                 if !is_uds_response_id(target_sa, seg.id) { | runtime gate; condition changes c
| 1105 |                                     continue; | support logic; impact depends on neighboring block and data flo
| 1106 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 1107 |                                 let sp = can_payload(&seg); | support logic; impact depends on neighboring bloc
| 1108 |                                 if sp.is_empty() { | runtime gate; condition changes can enable/disable critica
| 1109 |                                     continue; | support logic; impact depends on neighboring block and data flo
| 1110 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 1111 |                                 let t = (sp[0] >> 4) & 0x0F; | support logic; impact depends on neighboring blo
| 1112 |                                 if t != 0x2 { | runtime gate; condition changes can enable/disable critical pa
| 1113 |                                     continue; | support logic; impact depends on neighboring block and data flo
| 1114 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 1115 |                                 let sn_cf = sp[0] & 0x0F; | support logic; impact depends on neighboring block
| 1116 |                                 if sn_cf != expected_sn { | runtime gate; condition changes can enable/disable
| 1117 |                                     return Err(HwError::ParseError { | support logic; impact depends on neighbo
| 1118 |                                         cause: "ISO-TP CF sequence mismatch".to_string(), | support logic; impa
| 1119 |                                         raw_data: format!( | support logic; impact depends on neighboring block
| 1120 |                                             "expected_sn={expected_sn}, got_sn={sn_cf}" | support logic; impact
| 1121 |                                         ), | support logic; impact depends on neighboring block and data flow.
| 1122 |                                     }); | support logic; impact depends on neighboring block and data flow. |
| 1123 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 1124 |                                 expected_sn = (expected_sn + 1) & 0x0F; | support logic; impact depends on neig
| 1125 |                                 out.extend_from_slice(&sp[1..]); | support logic; impact depends on neighboring
| 1126 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 1127 |                             Ok(_) => {} | support logic; impact depends on neighboring block and data flow. |
| 1128 |                             Err(HwError::Timeout) => {} | support logic; impact depends on neighboring block an
| 1129 |                             Err(e) => return Err(e), | support logic; impact depends on neighboring block and c
| 1130 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1131 | (blank) | blank separator; no runtime behavior. |
| 1132 |                         if Instant::now() >= deadline { | runtime gate; condition changes can enable/disable cr
| 1133 |                             return Err(HwError::Timeout); | support logic; impact depends on neighboring block
| 1134 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1135 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1136 | (blank) | blank separator; no runtime behavior. |
| 1137 |                     out.truncate(total_len); | support logic; impact depends on neighboring block and data flow
| 1138 |                     if out.len() >= 3 && out[0] == 0x7F && out[2] == 0x78 { | runtime gate; condition changes c
| 1139 |                         deadline = Instant::now() | support logic; impact depends on neighboring block and data
| 1140 |                             + Duration::from_millis(cfg.uds_timeout_p2star_ms.max(1)); | protocol or I/O critic
| 1141 |                         continue; | support logic; impact depends on neighboring block and data flow. |
| 1142 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1143 |                     return Ok(out); | support logic; impact depends on neighboring block and data flow. |
| 1144 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1145 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1146 |             Ok(Frame::Serial(_)) => {} | support logic; impact depends on neighboring block and data flow. |
| 1147 |             Err(HwError::Timeout) => { | support logic; impact depends on neighboring block and data flow. |
| 1148 |                 if cfg.keep_channels_alive | runtime gate; condition changes can enable/disable critical paths
| 1149 |                     && cfg.can_interface.is_some() | support logic; impact depends on neighboring block and da
| 1150 |                     && last_keepalive.elapsed() | support logic; impact depends on neighboring block and data
| 1151 |                         >= Duration::from_millis(cfg.j1939_idle_guard_ms.max(250)) | support logic; impact depe
| 1152 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1153 |                     if let Some(sa) = target_sa { | runtime gate; condition changes can enable/disable critical
| 1154 |                         let _ = send_can_sf(adapter, cfg, sa, &[0x3E, 0x00]); | protocol or I/O critical; chang
| 1155 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1156 |                     last_keepalive = Instant::now(); | support logic; impact depends on neighboring block and c
| 1157 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1158 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1159 |             Err(e) => return Err(e), | support logic; impact depends on neighboring block and data flow. |
| 1160 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1161 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1162 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1163 | (blank) | blank separator; no runtime behavior. |
| 1164 | fn send_uds_and_wait( | function signature/body; changes affect API behavior and control-flow. |
| 1165 |     adapter: &mut dyn super::hw::HardwareInterface, | support logic; impact depends on neighboring block and da
| 1166 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 1167 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 1168 |     req: &[u8], | support logic; impact depends on neighboring block and data flow. |
| 1169 |     timeout: Duration, | support logic; impact depends on neighboring block and data flow. |
| 1170 | ) -> Result<Vec<u8>, HwError> { | support logic; impact depends on neighboring block and data flow. |
```

```
| 1171 |     if cfg.can_interface.is_some() { | runtime gate; condition changes can enable/disable critical paths. |
| 1172 |         let dst_sa = target_sa.unwrap_or(0x00); | support logic; impact depends on neighboring block and data f
| 1173 |         send_can_multiframe(adapter, cfg, dst_sa, req)?; | error propagation point; changes alter failure propa
| 1174 |         return recv_can_uds_response(adapter, cfg, target_sa, timeout); | protocol or I/O critical; changes may
| 1175 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1176 | (blank) | blank separator; no runtime behavior. |
| 1177 |     let frame = Frame::Serial(super::hw::SerialFrame { | support logic; impact depends on neighboring block and
| 1178 |         bytes: req.to_vec(), | support logic; impact depends on neighboring block and data flow. |
| 1179 |         protocol_hint: Some("uds-raw".into()), | protocol or I/O critical; changes may impact external integrat
| 1180 |         timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 1181 |     }); | support logic; impact depends on neighboring block and data flow. |
| 1182 |     send_frame_retry( | protocol or I/O critical; changes may impact external integration correctness. |
| 1183 |         adapter, | support logic; impact depends on neighboring block and data flow. |
| 1184 |         frame, | support logic; impact depends on neighboring block and data flow. |
| 1185 |         cfg.uds_retry_count, | protocol or I/O critical; changes may impact external integration correctness.
| 1186 |         cfg.write_retry_backoff_ms, | protocol or I/O critical; changes may impact external integration correc
| 1187 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1188 | (blank) | blank separator; no runtime behavior. |
| 1189 |     let deadline = Instant::now() + timeout; | support logic; impact depends on neighboring block and data flo
| 1190 |     while Instant::now() < deadline { | iteration logic; may alter timing, throughput, and safety constraints.
| 1191 |         match adapter.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 1192 |             Ok(Frame::Serial(sf)) => { | support logic; impact depends on neighboring block and data flow. |
| 1193 |                 if sf | runtime gate; condition changes can enable/disable critical paths. |
| 1194 |                     .protocol_hint | support logic; impact depends on neighboring block and data flow. |
| 1195 |                     .as_deref() | support logic; impact depends on neighboring block and data flow. |
| 1196 |                     .is_some_and(\|h\| !h.eq_ignore_ascii_case("uds-raw")) | protocol or I/O critical; changes
| 1197 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1198 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 1199 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1200 |                 if sf.bytes.is_empty() { | runtime gate; condition changes can enable/disable critical paths.
| 1201 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 1202 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1203 |                 if sf.bytes.len() >= 3 && sf.bytes[0] == 0x7F && sf.bytes[2] == 0x78 { | runtime gate; conditi
| 1204 |                     continue; | support logic; impact depends on neighboring block and data flow. |
| 1205 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1206 |                 return Ok(sf.bytes); | support logic; impact depends on neighboring block and data flow. |
| 1207 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1208 |             Ok(Frame::Can(_)) => {} | support logic; impact depends on neighboring block and data flow. |
| 1209 |             Err(HwError::Timeout) => {} | support logic; impact depends on neighboring block and data flow. |
| 1210 |             Err(e) => return Err(e), | support logic; impact depends on neighboring block and data flow. |
| 1211 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1212 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1213 |     Err(HwError::Timeout) | support logic; impact depends on neighboring block and data flow. |
| 1214 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1215 | (blank) | blank separator; no runtime behavior. |
| 1216 | fn check_live_channel_policy(cfg: &HwConfig) -> Result<(), HwError> { | function signature/body; changes affec
| 1217 |     if cfg.ethernet_probe.is_some() && cfg.can_interface.is_none() { | runtime gate; condition changes can enal
| 1218 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 1219 |             "ethernet/internet checks require CAN/J1939 active (--can-if) to keep channel continuity" | suppor
| 1220 |                 .to_string(), | support logic; impact depends on neighboring block and data flow. |
| 1221 |         )); | support logic; impact depends on neighboring block and data flow. |
| 1222 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1223 | (blank) | blank separator; no runtime behavior. |
| 1224 |     if let Some(target) = &cfg.ethernet_probe { | runtime gate; condition changes can enable/disable critical
| 1225 |         let mut addrs = target | support logic; impact depends on neighboring block and data flow. |
| 1226 |             .to_socket_addrs() | support logic; impact depends on neighboring block and data flow. |
| 1227 |             .map_err(\|e\| HwError::Unknown(format!("invalid --eth-probe target: {e}")))?; | error propagation
| 1228 |         let addr = addrs | support logic; impact depends on neighboring block and data flow. |
| 1229 |             .next() | support logic; impact depends on neighboring block and data flow. |
| 1230 |             .ok_or_else(\|\| HwError::Unknown("--eth-probe resolved no socket address".to_string()))?; | error
| 1231 |         TcpStream::connect_timeout(&addr, Duration::from_millis(350)).map_err(\|e\| { | support logic; impact
| 1232 |             HwError::Unknown(format!( | support logic; impact depends on neighboring block and data flow. |
| 1233 |                 "ethernet/internet check failed ({target}): {e}; preserving J1939 stability" | support logic;
| 1234 |             )) | support logic; impact depends on neighboring block and data flow. |
| 1235 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 1236 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1237 | (blank) | blank separator; no runtime behavior. |
| 1238 |     Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 1239 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1240 | (blank) | blank separator; no runtime behavior. |
| 1241 | pub fn live_clean_ecm(cfg: &HwConfig, target_sa: Option<u8>) -> Result<(), HwError> { | function signature/body
| 1242 |     if cfg.mode != HwMode::Live { | runtime gate; condition changes can enable/disable critical paths. |
| 1243 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 1244 |             "clean requires --hw-mode=live".to_string(), | support logic; impact depends on neighboring block
| 1245 |         )); | support logic; impact depends on neighboring block and data flow. |
| 1246 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 1247 |     if !cfg.write_effectively_enabled() { | runtime gate; condition changes can enable/disable critical paths.
| 1248 |         return Err(HwError::WriteBlockedAllowlist); | support logic; impact depends on neighboring block and da
| 1249 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1250 |     check_live_channel_policy(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 1251 | (blank) | blank separator; no runtime behavior. |
| 1252 |     let mut adapter = super::hw::open_real_adapter(cfg)?; | error propagation point; changes alter failure prop
| 1253 |     adapter.init(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 1254 | (blank) | blank separator; no runtime behavior. |
| 1255 |     let session_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration co
| 1256 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1257 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1258 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1259 |         &[0x10, 0x02], | support logic; impact depends on neighboring block and data flow. |
| 1260 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1261 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1262 |     ensure_positive_response(&session_rsp, 0x10, "clean::session", cfg, target_sa)?; | error propagation point
| 1263 | (blank) | blank separator; no runtime behavior. |
| 1264 |     let clear_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration cor
| 1265 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1266 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1267 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1268 |         &[0x14, 0xFF, 0xFF, 0xFF], | support logic; impact depends on neighboring block and data flow. |
| 1269 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1270 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1271 |     ensure_positive_response(&clear_rsp, 0x14, "clean::clear_dtc", cfg, target_sa)?; | error propagation point
| 1272 | (blank) | blank separator; no runtime behavior. |
| 1273 |     let routine_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration co
| 1274 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1275 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1276 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1277 |         &[0x31, 0x01, 0xDF, 0x04], | support logic; impact depends on neighboring block and data flow. |
| 1278 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1279 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1280 |     ensure_positive_response(&routine_rsp, 0x31, "clean::routine", cfg, target_sa)?; | error propagation point
| 1281 | (blank) | blank separator; no runtime behavior. |
| 1282 |     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 1283 |     Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 1284 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1285 | (blank) | blank separator; no runtime behavior. |
| 1286 | pub fn live_flash_ecm_firmware( | function signature/body; changes affect API behavior and control-flow. |
| 1287 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 1288 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 1289 |     firmware: &[u8], | support logic; impact depends on neighboring block and data flow. |
| 1290 | ) -> Result<FlashSummary, HwError> { | protocol or I/O critical; changes may impact external integration corre
| 1291 |     if cfg.mode != HwMode::Live { | runtime gate; condition changes can enable/disable critical paths. |
| 1292 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 1293 |             "flash requires --hw-mode=live".to_string(), | protocol or I/O critical; changes may impact externa
| 1294 |         )); | support logic; impact depends on neighboring block and data flow. |
| 1295 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1296 |     if firmware.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 1297 |         return Err(HwError::Unknown("firmware payload is empty".to_string())); | support logic; impact depends
| 1298 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1299 |     if !cfg.write_effectively_enabled() { | runtime gate; condition changes can enable/disable critical paths.
| 1300 |         return Err(HwError::WriteBlockedAllowlist); | support logic; impact depends on neighboring block and da
| 1301 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1302 |     check_live_channel_policy(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 1303 |     let flash_start_ms = now_ms(); | protocol or I/O critical; changes may impact external integration correctn
| 1304 | (blank) | blank separator; no runtime behavior. |
| 1305 |     append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 1306 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1307 |         "flash::start", | protocol or I/O critical; changes may impact external integration correctness. |
| 1308 |         FlashAuditStatus::Ok, | protocol or I/O critical; changes may impact external integration correctness.
| 1309 |         format!("payload_bytes={}", firmware.len()), | support logic; impact depends on neighboring block and
| 1310 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1311 |     ); | support logic; impact depends on neighboring block and data flow. |
| 1312 | (blank) | blank separator; no runtime behavior. |
| 1313 |     let mut adapter = super::hw::open_real_adapter(cfg)?; | error propagation point; changes alter failure prop
| 1314 |     adapter.init(cfg)?; | error propagation point; changes alter failure propagation semantics. |
| 1315 |     let firmware_crc = crc32fast::hash(firmware); | support logic; impact depends on neighboring block and data
| 1316 |     let mut transfer_data_blocks_attempted = 0usize; | support logic; impact depends on neighboring block and
| 1317 |     let mut transfer_data_blocks_acked = 0usize; | support logic; impact depends on neighboring block and data
| 1318 |     let wait_frame_count = 0u32; | support logic; impact depends on neighboring block and data flow. |
| 1319 |     let sequence_error_count = 0u32; | support logic; impact depends on neighboring block and data flow. |
| 1320 |     let flowcontrol_timeout_count = 0u32; | support logic; impact depends on neighboring block and data flow.
| 1321 | (blank) | blank separator; no runtime behavior. |
| 1322 |     let preflight = run_flash_preflight(&mut *adapter, cfg, target_sa)?; | error propagation point; changes al
```

```
| 1323 |     append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 1324 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1325 |         "flash::preflight", | protocol or I/O critical; changes may impact external integration correctness. |
| 1326 |         FlashAuditStatus::Ok, | protocol or I/O critical; changes may impact external integration correctness.
| 1327 |         format!( | support logic; impact depends on neighboring block and data flow. |
| 1328 |             "supply_v={:.2}, vin={}, sw={}, hw={}, cal={}, serial={}, fp_present={}", | support logic; impact c
| 1329 |             preflight.supply_voltage_v, | support logic; impact depends on neighboring block and data flow. |
| 1330 |             preflight.identity.vin, | support logic; impact depends on neighboring block and data flow. |
| 1331 |             preflight.identity.sw_version, | support logic; impact depends on neighboring block and data flow.
| 1332 |             preflight.identity.hw_version, | support logic; impact depends on neighboring block and data flow.
| 1333 |             preflight.identity.calibration_id, | support logic; impact depends on neighboring block and data fl
| 1334 |             preflight.identity.ecu_serial, | support logic; impact depends on neighboring block and data flow.
| 1335 |             preflight.identity.fingerprint.is_some() | support logic; impact depends on neighboring block and c
| 1336 |         ), | support logic; impact depends on neighboring block and data flow. |
| 1337 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1338 |     ); | support logic; impact depends on neighboring block and data flow. |
| 1339 | (blank) | blank separator; no runtime behavior. |
| 1340 |     let default_session_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integr
| 1341 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1342 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1343 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1344 |         &[0x10, 0x02], | support logic; impact depends on neighboring block and data flow. |
| 1345 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1346 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1347 |     ensure_positive_response( | support logic; impact depends on neighboring block and data flow. |
| 1348 |         &default_session_rsp, | support logic; impact depends on neighboring block and data flow. |
| 1349 |         0x10, | support logic; impact depends on neighboring block and data flow. |
| 1350 |         "flash::default_session", | protocol or I/O critical; changes may impact external integration correctne
| 1351 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1352 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1353 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1354 | (blank) | blank separator; no runtime behavior. |
| 1355 |     let seed = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration correctne
| 1356 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1357 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1358 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1359 |         &[0x27, 0x05], | support logic; impact depends on neighboring block and data flow. |
| 1360 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1361 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1362 |     ensure_positive_response(&seed, 0x27, "flash::security_seed", cfg, target_sa)?; | error propagation point;
| 1363 |     if seed.len() < 6 { | runtime gate; condition changes can enable/disable critical paths. |
| 1364 |         append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 1365 |             cfg, | support logic; impact depends on neighboring block and data flow. |
| 1366 |             "flash::security_seed", | protocol or I/O critical; changes may impact external integration correct
| 1367 |             FlashAuditStatus::Failed, | protocol or I/O critical; changes may impact external integration corr
| 1368 |             format!("short seed response len={}", seed.len()), | support logic; impact depends on neighboring b
| 1369 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1370 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1371 |         adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 1372 |         return Err(HwError::Unknown("security seed response invalid".to_string())); | support logic; impact dep
| 1373 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1374 |     let seed_u32 = ((seed[2] as u32) << 24) | support logic; impact depends on neighboring block and data flow
| 1375 |         \| ((seed[3] as u32) << 16) | support logic; impact depends on neighboring block and data flow. |
| 1376 |         \| ((seed[4] as u32) << 8) | support logic; impact depends on neighboring block and data flow. |
| 1377 |         \| seed[5] as u32; | support logic; impact depends on neighboring block and data flow. |
| 1378 |     let key = seed_u32 ^ 0xDEADBEEF; | support logic; impact depends on neighboring block and data flow. |
| 1379 |     let key_req = [ | support logic; impact depends on neighboring block and data flow. |
| 1380 |         0x27, | support logic; impact depends on neighboring block and data flow. |
| 1381 |         0x06, | support logic; impact depends on neighboring block and data flow. |
| 1382 |         ((key >> 24) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1383 |         ((key >> 16) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1384 |         ((key >> 8) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1385 |         (key & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1386 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 1387 |     let unlock_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration co
| 1388 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1389 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1390 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1391 |         &key_req, | support logic; impact depends on neighboring block and data flow. |
| 1392 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1393 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1394 |     ensure_positive_response(&unlock_rsp, 0x27, "flash::security_unlock", cfg, target_sa)?; | error propagatio
| 1395 | (blank) | blank separator; no runtime behavior. |
| 1396 |     let prog_session_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integrati
| 1397 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1398 |         cfg, | support logic; impact depends on neighboring block and data flow. |
```

```
| 1399 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1400 |         &[0x10, 0x03], | support logic; impact depends on neighboring block and data flow. |
| 1401 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1402 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1403 |     ensure_positive_response( | support logic; impact depends on neighboring block and data flow. |
| 1404 |         &prog_session_rsp, | support logic; impact depends on neighboring block and data flow. |
| 1405 |         0x10, | support logic; impact depends on neighboring block and data flow. |
| 1406 |         "flash::programming_session", | protocol or I/O critical; changes may impact external integration corre
| 1407 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1408 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1409 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1410 | (blank) | blank separator; no runtime behavior. |
| 1411 |     let total_len = firmware.len() as u32; | support logic; impact depends on neighboring block and data flow.
| 1412 |     let req_dl = [ | support logic; impact depends on neighboring block and data flow. |
| 1413 |         0x34, | support logic; impact depends on neighboring block and data flow. |
| 1414 |         0x00, | support logic; impact depends on neighboring block and data flow. |
| 1415 |         0x00, | support logic; impact depends on neighboring block and data flow. |
| 1416 |         0x00, | support logic; impact depends on neighboring block and data flow. |
| 1417 |         0x00, | support logic; impact depends on neighboring block and data flow. |
| 1418 |         0x00, | support logic; impact depends on neighboring block and data flow. |
| 1419 |         ((total_len >> 24) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow.
| 1420 |         ((total_len >> 16) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow.
| 1421 |         ((total_len >> 8) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1422 |         (total_len & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1423 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 1424 |     let dl_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration correc
| 1425 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1426 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1427 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1428 |         &req_dl, | support logic; impact depends on neighboring block and data flow. |
| 1429 |         Duration::from_millis(cfg.uds_timeout_p2_ms.max(1)), | protocol or I/O critical; changes may impact ext
| 1430 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1431 |     ensure_positive_response(&dl_rsp, 0x34, "flash::request_download", cfg, target_sa)?; | error propagation po
| 1432 | (blank) | blank separator; no runtime behavior. |
| 1433 |     let mut max_block_payload = 127usize; | support logic; impact depends on neighboring block and data flow.
| 1434 |     if dl_rsp.len() >= 4 && dl_rsp[0] == 0x74 { | runtime gate; condition changes can enable/disable critical p
| 1435 |         let mbl = ((dl_rsp[2] as u16) << 8) \| dl_rsp[3] as u16; | support logic; impact depends on neighboring
| 1436 |         if mbl > 2 { | runtime gate; condition changes can enable/disable critical paths. |
| 1437 |             max_block_payload = (mbl as usize).saturating_sub(2).max(1); | support logic; impact depends on nei
| 1438 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1439 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1440 |     max_block_payload = max_block_payload.min(2048); | support logic; impact depends on neighboring block and d
| 1441 | (blank) | blank separator; no runtime behavior. |
| 1442 |     let mut block = 1u8; | support logic; impact depends on neighboring block and data flow. |
| 1443 |     let mut blocks = 0usize; | support logic; impact depends on neighboring block and data flow. |
| 1444 |     for chunk in firmware.chunks(max_block_payload) { | iteration logic; may alter timing, throughput, and safe
| 1445 |         let mut req = Vec::with_capacity(chunk.len() + 2); | support logic; impact depends on neighboring bloc
| 1446 |         req.push(0x36); | support logic; impact depends on neighboring block and data flow. |
| 1447 |         req.push(block); | support logic; impact depends on neighboring block and data flow. |
| 1448 |         req.extend_from_slice(chunk); | support logic; impact depends on neighboring block and data flow. |
| 1449 |         transfer_data_blocks_attempted += 1; | support logic; impact depends on neighboring block and data flow
| 1450 |         let ack = match send_uds_and_wait( | protocol or I/O critical; changes may impact external integration
| 1451 |             &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1452 |             cfg, | support logic; impact depends on neighboring block and data flow. |
| 1453 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1454 |             &req, | support logic; impact depends on neighboring block and data flow. |
| 1455 |             Duration::from_millis(cfg.uds_timeout_p2star_ms.max(1)), | protocol or I/O critical; changes may in
| 1456 |         ) { | support logic; impact depends on neighboring block and data flow. |
| 1457 |             Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 1458 |             Err(HwError::Timeout) => { | support logic; impact depends on neighboring block and data flow. |
| 1459 |                 append_flash_audit( | protocol or I/O critical; changes may impact external integration correc
| 1460 |                     cfg, | support logic; impact depends on neighboring block and data flow. |
| 1461 |                     "flash::transfer_data", | protocol or I/O critical; changes may impact external integratio
| 1462 |                     FlashAuditStatus::Failed, | protocol or I/O critical; changes may impact external integrati
| 1463 |                     format!("timeout waiting transfer ack block={block}"), | support logic; impact depends on n
| 1464 |                     target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1465 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 1466 |                 adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 1467 |                 return Err(HwError::Timeout); | support logic; impact depends on neighboring block and data flo
| 1468 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1469 |             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 1470 |                 adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 1471 |                 return Err(e); | support logic; impact depends on neighboring block and data flow. |
| 1472 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1473 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1474 |         ensure_positive_response(&ack, 0x36, "flash::transfer_data", cfg, target_sa)?; | error propagation poi
```

```
| 1475 |         if ack.len() < 2 \|\| ack[1] != block { | runtime gate; condition changes can enable/disable critical p
| 1476 |             append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness
| 1477 |                 cfg, | support logic; impact depends on neighboring block and data flow. |
| 1478 |                 "flash::transfer_data", | protocol or I/O critical; changes may impact external integration co
| 1479 |                 FlashAuditStatus::Failed, | protocol or I/O critical; changes may impact external integration c
| 1480 |                 format!("ack block mismatch expected={block} got={}", ack.get(1).copied().unwrap_or(0xFF)), | s
| 1481 |                 target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1482 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1483 |             adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 1484 |             return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring block and data
| 1485 |                 "transfer ack mismatch for block {block}" | support logic; impact depends on neighboring block
| 1486 |             ))); | support logic; impact depends on neighboring block and data flow. |
| 1487 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1488 |         transfer_data_blocks_acked += 1; | support logic; impact depends on neighboring block and data flow. |
| 1489 |         block = block.wrapping_add(1); | support logic; impact depends on neighboring block and data flow. |
| 1490 |         blocks += 1; | support logic; impact depends on neighboring block and data flow. |
| 1491 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1492 | (blank) | blank separator; no runtime behavior. |
| 1493 |     let exit_req_crc = [ | support logic; impact depends on neighboring block and data flow. |
| 1494 |         0x37, | support logic; impact depends on neighboring block and data flow. |
| 1495 |         ((firmware_crc >> 24) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flo
| 1496 |         ((firmware_crc >> 16) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flo
| 1497 |         ((firmware_crc >> 8) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow
| 1498 |         (firmware_crc & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1499 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 1500 |     let exit = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration correctne
| 1501 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1502 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1503 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1504 |         &exit_req_crc, | support logic; impact depends on neighboring block and data flow. |
| 1505 |         Duration::from_millis(cfg.uds_timeout_p2star_ms.max(1)), | protocol or I/O critical; changes may impact
| 1506 |     ) | support logic; impact depends on neighboring block and data flow. |
| 1507 |     .or_else(\|_\| { | support logic; impact depends on neighboring block and data flow. |
| 1508 |         send_uds_and_wait( | protocol or I/O critical; changes may impact external integration correctness. |
| 1509 |             &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1510 |             cfg, | support logic; impact depends on neighboring block and data flow. |
| 1511 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1512 |             &[0x37, 0x00], | support logic; impact depends on neighboring block and data flow. |
| 1513 |             Duration::from_millis(cfg.uds_timeout_p2star_ms.max(1)), | protocol or I/O critical; changes may im
| 1514 |         ) | support logic; impact depends on neighboring block and data flow. |
| 1515 |     })?; | error propagation point; changes alter failure propagation semantics. |
| 1516 |     ensure_positive_response(&exit, 0x37, "flash::transfer_exit", cfg, target_sa)?; | error propagation point;
| 1517 |     if exit.first().copied() != Some(0x77) { | runtime gate; condition changes can enable/disable critical path
| 1518 |         append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 1519 |             cfg, | support logic; impact depends on neighboring block and data flow. |
| 1520 |             "flash::transfer_exit", | protocol or I/O critical; changes may impact external integration correct
| 1521 |             FlashAuditStatus::Failed, | protocol or I/O critical; changes may impact external integration corre
| 1522 |             "transfer exit rejected", | support logic; impact depends on neighboring block and data flow. |
| 1523 |             target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1524 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1525 |         adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
| 1526 |         return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 1527 |             "transfer exit rejected during integrity verification".to_string(), | support logic; impact depends
| 1528 |         )); | support logic; impact depends on neighboring block and data flow. |
| 1529 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1530 | (blank) | blank separator; no runtime behavior. |
| 1531 |     let verify_req = [ | support logic; impact depends on neighboring block and data flow. |
| 1532 |         0x31, | support logic; impact depends on neighboring block and data flow. |
| 1533 |         0x01, | support logic; impact depends on neighboring block and data flow. |
| 1534 |         0xF1, | support logic; impact depends on neighboring block and data flow. |
| 1535 |         0x90, | support logic; impact depends on neighboring block and data flow. |
| 1536 |         ((firmware_crc >> 24) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flo
| 1537 |         ((firmware_crc >> 16) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flo
| 1538 |         ((firmware_crc >> 8) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow
| 1539 |         (firmware_crc & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 1540 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 1541 |     let verify_rsp = send_uds_and_wait( | protocol or I/O critical; changes may impact external integration co
| 1542 |         &mut *adapter, | support logic; impact depends on neighboring block and data flow. |
| 1543 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1544 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1545 |         &verify_req, | support logic; impact depends on neighboring block and data flow. |
| 1546 |         Duration::from_millis(cfg.uds_timeout_p2star_ms.max(1)), | protocol or I/O critical; changes may impact
| 1547 |     )?; | error propagation point; changes alter failure propagation semantics. |
| 1548 |     ensure_positive_response(&verify_rsp, 0x31, "flash::verify_routine", cfg, target_sa)?; | error propagation
| 1549 | (blank) | blank separator; no runtime behavior. |
| 1550 |     adapter.close()?; | error propagation point; changes alter failure propagation semantics. |
```

```
| 1551 | (blank) | blank separator; no runtime behavior. |
| 1552 |     append_flash_audit( | protocol or I/O critical; changes may impact external integration correctness. |
| 1553 |         cfg, | support logic; impact depends on neighboring block and data flow. |
| 1554 |         "flash::complete", | protocol or I/O critical; changes may impact external integration correctness. |
| 1555 |         FlashAuditStatus::Ok, | protocol or I/O critical; changes may impact external integration correctness.
| 1556 |         format!("bytes={} blocks={} crc32=0x{firmware_crc:08X}", firmware.len(), blocks), | support logic; impa
| 1557 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 1558 |     ); | support logic; impact depends on neighboring block and data flow. |
| 1559 | (blank) | blank separator; no runtime behavior. |
| 1560 |     let bridge_diag = read_bridge_diag_since(cfg, flash_start_ms); | protocol or I/O critical; changes may impa
| 1561 | (blank) | blank separator; no runtime behavior. |
| 1562 |     Ok(FlashSummary { | protocol or I/O critical; changes may impact external integration correctness. |
| 1563 |         bytes_sent: firmware.len(), | support logic; impact depends on neighboring block and data flow. |
| 1564 |         blocks_sent: blocks, | support logic; impact depends on neighboring block and data flow. |
| 1565 |         crc32: firmware_crc, | support logic; impact depends on neighboring block and data flow. |
| 1566 |         supply_voltage_v: preflight.supply_voltage_v, | support logic; impact depends on neighboring block and
| 1567 |         vin: preflight.identity.vin, | support logic; impact depends on neighboring block and data flow. |
| 1568 |         sw_version: preflight.identity.sw_version, | support logic; impact depends on neighboring block and da
| 1569 |         hw_version: preflight.identity.hw_version, | support logic; impact depends on neighboring block and da
| 1570 |         calibration_id: preflight.identity.calibration_id, | support logic; impact depends on neighboring bloc
| 1571 |         ecu_serial: preflight.identity.ecu_serial, | support logic; impact depends on neighboring block and da
| 1572 |         transport_diagnostics: FlashTransportDiagnostics { | protocol or I/O critical; changes may impact exte
| 1573 |             security_seed_positive: true, | support logic; impact depends on neighboring block and data flow.
| 1574 |             security_unlock_positive: true, | support logic; impact depends on neighboring block and data flow
| 1575 |             request_download_positive: true, | support logic; impact depends on neighboring block and data flo
| 1576 |             transfer_data_blocks_attempted, | support logic; impact depends on neighboring block and data flow
| 1577 |             transfer_data_blocks_acked, | support logic; impact depends on neighboring block and data flow. |
| 1578 |             request_transfer_exit_positive: true, | support logic; impact depends on neighboring block and data
| 1579 |             fc_blocksize_seen: bridge_diag.fc_blocksize_seen.or(Some(cfg.uds_block_size)), | protocol or I/O c
| 1580 |             fc_stmin_seen_ms: bridge_diag.fc_stmin_seen_ms.or(Some(cfg.uds_st_min_ms)), | protocol or I/O crit
| 1581 |             wait_frame_count: wait_frame_count.saturating_add(bridge_diag.wait_frame_count), | support logic; i
| 1582 |             sequence_error_count: sequence_error_count | support logic; impact depends on neighboring block an
| 1583 |                 .saturating_add(bridge_diag.sequence_error_count), | support logic; impact depends on neighbor
| 1584 |             flowcontrol_timeout_count: flowcontrol_timeout_count | support logic; impact depends on neighborin
| 1585 |                 .saturating_add(bridge_diag.flowcontrol_timeout_count), | support logic; impact depends on nei
| 1586 |         }, | support logic; impact depends on neighboring block and data flow. |
| 1587 |     }) | support logic; impact depends on neighboring block and data flow. |
| 1588 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1589 | (blank) | blank separator; no runtime behavior. |
| 1590 | fn read_bridge_diag_since(cfg: &HwConfig, since_ms: u64) -> BridgeDiagAggregate { | function signature/body; c
| 1591 |     let path = cfg.log_dir.join("cat_comm_bridge_diag.jsonl"); | support logic; impact depends on neighboring
| 1592 |     let content = match fs::read_to_string(path) { | support logic; impact depends on neighboring block and da
| 1593 |         Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 1594 |         Err(_) => return BridgeDiagAggregate::default(), | support logic; impact depends on neighboring block
| 1595 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1596 | (blank) | blank separator; no runtime behavior. |
| 1597 |     let mut agg = BridgeDiagAggregate::default(); | support logic; impact depends on neighboring block and data
| 1598 |     for line in content.lines() { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1599 |         let Ok(r) = serde_json::from_str::<BridgeDiagLine>(line) else { | support logic; impact depends on nei
| 1600 |             continue; | support logic; impact depends on neighboring block and data flow. |
| 1601 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1602 |         if r.ts_ms < since_ms { | runtime gate; condition changes can enable/disable critical paths. |
| 1603 |             continue; | support logic; impact depends on neighboring block and data flow. |
| 1604 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1605 |         agg.wait_frame_count = agg | support logic; impact depends on neighboring block and data flow. |
| 1606 |             .wait_frame_count | support logic; impact depends on neighboring block and data flow. |
| 1607 |             .saturating_add(r.wait_frame_count_delta.unwrap_or(0)); | support logic; impact depends on neighbo
| 1608 |         agg.sequence_error_count = agg | support logic; impact depends on neighboring block and data flow. |
| 1609 |             .sequence_error_count | support logic; impact depends on neighboring block and data flow. |
| 1610 |             .saturating_add(r.sequence_error_count_delta.unwrap_or(0)); | support logic; impact depends on nei
| 1611 |         agg.flowcontrol_timeout_count = agg | support logic; impact depends on neighboring block and data flow
| 1612 |             .flowcontrol_timeout_count | support logic; impact depends on neighboring block and data flow. |
| 1613 |             .saturating_add(r.flowcontrol_timeout_count_delta.unwrap_or(0)); | support logic; impact depends o
| 1614 |         if r.fc_blocksize.is_some() { | runtime gate; condition changes can enable/disable critical paths. |
| 1615 |             agg.fc_blocksize_seen = r.fc_blocksize; | support logic; impact depends on neighboring block and d
| 1616 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1617 |         if r.fc_stmin_ms.is_some() { | runtime gate; condition changes can enable/disable critical paths. |
| 1618 |             agg.fc_stmin_seen_ms = r.fc_stmin_ms; | support logic; impact depends on neighboring block and data
| 1619 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1620 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1621 |     agg | support logic; impact depends on neighboring block and data flow. |
| 1622 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1623 | (blank) | blank separator; no runtime behavior. |
| 1624 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 1625 | mod tests { | module declaration; changing can break namespace graph. |
| 1626 |     use super::ensure_positive_response; | import wiring; changing path/name can break compilation and module
```

```
| 1627 |     use super::is_uds_response_id; | import wiring; changing path/name can break compilation and module linkage
| 1628 |     use crate::io::hw::HwConfig; | import wiring; changing path/name can break compilation and module linkage.
| 1629 | (blank) | blank separator; no runtime behavior. |
| 1630 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1631 |     fn uds_response_id_matches_expected_target() { | function signature/body; changes affect API behavior and c
| 1632 |         // PF=0xDA, PS=0xF9, SA=0x00 | comment/documentation; changing affects understanding and intent traceab
| 1633 |         let id = 0x18DAF900u32; | support logic; impact depends on neighboring block and data flow. |
| 1634 |         assert!(is_uds_response_id(Some(0x00), id)); | protocol or I/O critical; changes may impact external in
| 1635 |         assert!(is_uds_response_id(None, id)); | protocol or I/O critical; changes may impact external integrat
| 1636 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1637 | (blank) | blank separator; no runtime behavior. |
| 1638 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1639 |     fn uds_response_id_rejects_wrong_pf_ps_or_sa() { | function signature/body; changes affect API behavior and
| 1640 |         let wrong_pf = 0x18EAF900u32; | support logic; impact depends on neighboring block and data flow. |
| 1641 |         let wrong_ps = 0x18DA1200u32; | support logic; impact depends on neighboring block and data flow. |
| 1642 |         let wrong_sa = 0x18DAF901u32; | support logic; impact depends on neighboring block and data flow. |
| 1643 | (blank) | blank separator; no runtime behavior. |
| 1644 |         assert!(!is_uds_response_id(Some(0x00), wrong_pf)); | protocol or I/O critical; changes may impact exte
| 1645 |         assert!(!is_uds_response_id(Some(0x00), wrong_ps)); | protocol or I/O critical; changes may impact exte
| 1646 |         assert!(!is_uds_response_id(Some(0x00), wrong_sa)); | protocol or I/O critical; changes may impact exte
| 1647 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1648 | (blank) | blank separator; no runtime behavior. |
| 1649 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1650 |     fn ensure_positive_response_accepts_expected_sid() { | function signature/body; changes affect API behavior
| 1651 |         let cfg = HwConfig::default(); | support logic; impact depends on neighboring block and data flow. |
| 1652 |         let rsp = [0x50, 0x02, 0x00, 0x00]; | support logic; impact depends on neighboring block and data flow
| 1653 |         let result = ensure_positive_response(&rsp, 0x10, "test::session", &cfg, Some(0x00)); | support logic;
| 1654 |         assert!(result.is_ok()); | support logic; impact depends on neighboring block and data flow. |
| 1655 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1656 | (blank) | blank separator; no runtime behavior. |
| 1657 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1658 |     fn ensure_positive_response_rejects_negative_response_code() { | function signature/body; changes affect AP
| 1659 |         let cfg = HwConfig::default(); | support logic; impact depends on neighboring block and data flow. |
| 1660 |         let rsp = [0x7F, 0x10, 0x22]; | support logic; impact depends on neighboring block and data flow. |
| 1661 |         let result = ensure_positive_response(&rsp, 0x10, "test::session", &cfg, Some(0x00)); | support logic;
| 1662 |         assert!(result.is_err()); | support logic; impact depends on neighboring block and data flow. |
| 1663 |         let msg = format!("{}", result.err().expect("error expected")); | panic boundary; edits alter crash bel
| 1664 |         assert!(msg.contains("NRC=0x22"), "unexpected error: {msg}"); | support logic; impact depends on neighb
| 1665 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1666 | (blank) | blank separator; no runtime behavior. |
| 1667 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1668 |     fn ensure_positive_response_rejects_unexpected_sid() { | function signature/body; changes affect API behavi
| 1669 |         let cfg = HwConfig::default(); | support logic; impact depends on neighboring block and data flow. |
| 1670 |         let rsp = [0x62, 0xF1, 0x90]; | support logic; impact depends on neighboring block and data flow. |
| 1671 |         let result = ensure_positive_response(&rsp, 0x10, "test::session", &cfg, Some(0x00)); | support logic;
| 1672 |         assert!(result.is_err()); | support logic; impact depends on neighboring block and data flow. |
| 1673 |         let msg = format!("{}", result.err().expect("error expected")); | panic boundary; edits alter crash bel
| 1674 |         assert!(msg.contains("expected=0x50"), "unexpected error: {msg}"); | support logic; impact depends on n
| 1675 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1676 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/io/metrics.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 48 | Non-empty: 40 | Symbol count: 10

### Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| on_rx_can | 1 |
| on_rx_serial | 1 |
| on_tx_can_allowed | 1 |
| on_tx_serial_allowed | 1 |
| on_tx_blocked | 1 |
| on_rate_limited | 1 |
| on_read_error | 1 |
| on_write_error | 1 |

### Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|

```
| 2 | struct | HwMetrics | data/schema coupling and match/serde break risk |
| 14 | impl | HwMetrics | method behavior and trait semantics |
| 15 | fn | on_rx_can | API and behavior coupling to callers/implementers |
| 20 | fn | on_rx_serial | API and behavior coupling to callers/implementers |
| 25 | fn | on_tx_can_allowed | API and behavior coupling to callers/implementers |
| 29 | fn | on_tx_serial_allowed | API and behavior coupling to callers/implementers |
| 33 | fn | on_tx_blocked | API and behavior coupling to callers/implementers |
| 37 | fn | on_rate_limited | API and behavior coupling to callers/implementers |
| 41 | fn | on_read_error | API and behavior coupling to callers/implementers |
| 45 | fn | on_write_error | API and behavior coupling to callers/implementers |
```

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #[derive(Debug, Default, Clone)] | comment/documentation; changing affects understanding and intent traceability.
| 2 | pub struct HwMetrics { | data layout contract; field changes can break callers/serde/tests. |
| 3 |     pub hw_rx_frames_total_can: u64, | support logic; impact depends on neighboring block and data flow. |
| 4 |     pub hw_rx_frames_total_serial: u64, | support logic; impact depends on neighboring block and data flow. |
| 5 |     pub hw_tx_frames_total_can_allowed: u64, | support logic; impact depends on neighboring block and data flow.
| 6 |     pub hw_tx_frames_total_serial_allowed: u64, | support logic; impact depends on neighboring block and data flo
| 7 |     pub hw_tx_frames_total_blocked: u64, | support logic; impact depends on neighboring block and data flow. |
| 8 |     pub hw_rate_limited_total: u64, | support logic; impact depends on neighboring block and data flow. |
| 9 |     pub hw_read_errors_total: u64, | support logic; impact depends on neighboring block and data flow. |
| 10 |     pub hw_write_errors_total: u64, | protocol or I/O critical; changes may impact external integration correctne
| 11 |     pub hw_last_rx_timestamp_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 12 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 | impl HwMetrics { | implementation binding; behavior semantics and trait conformance live here. |
| 15 |     pub fn on_rx_can(&mut self, ts_ms: u64) { | function signature/body; changes affect API behavior and control
| 16 |         self.hw_rx_frames_total_can += 1; | support logic; impact depends on neighboring block and data flow. |
| 17 |         self.hw_last_rx_timestamp_ms = ts_ms; | support logic; impact depends on neighboring block and data flow
| 18 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 19 | (blank) | blank separator; no runtime behavior. |
| 20 |     pub fn on_rx_serial(&mut self, ts_ms: u64) { | function signature/body; changes affect API behavior and cont
| 21 |         self.hw_rx_frames_total_serial += 1; | support logic; impact depends on neighboring block and data flow.
| 22 |         self.hw_last_rx_timestamp_ms = ts_ms; | support logic; impact depends on neighboring block and data flow
| 23 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 |     pub fn on_tx_can_allowed(&mut self) { | function signature/body; changes affect API behavior and control-flo
| 26 |         self.hw_tx_frames_total_can_allowed += 1; | support logic; impact depends on neighboring block and data
| 27 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 28 | (blank) | blank separator; no runtime behavior. |
| 29 |     pub fn on_tx_serial_allowed(&mut self) { | function signature/body; changes affect API behavior and control-
| 30 |         self.hw_tx_frames_total_serial_allowed += 1; | support logic; impact depends on neighboring block and da
| 31 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 32 | (blank) | blank separator; no runtime behavior. |
| 33 |     pub fn on_tx_blocked(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 34 |         self.hw_tx_frames_total_blocked += 1; | support logic; impact depends on neighboring block and data flow
| 35 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 36 | (blank) | blank separator; no runtime behavior. |
| 37 |     pub fn on_rate_limited(&mut self) { | function signature/body; changes affect API behavior and control-flow.
| 38 |         self.hw_rate_limited_total += 1; | support logic; impact depends on neighboring block and data flow. |
| 39 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 40 | (blank) | blank separator; no runtime behavior. |
| 41 |     pub fn on_read_error(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 42 |         self.hw_read_errors_total += 1; | support logic; impact depends on neighboring block and data flow. |
| 43 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 44 | (blank) | blank separator; no runtime behavior. |
| 45 |     pub fn on_write_error(&mut self) { | function signature/body; changes affect API behavior and control-flow.
| 46 |         self.hw_write_errors_total += 1; | protocol or I/O critical; changes may impact external integration cor
| 47 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/io/mock.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 187 | Non-empty: 160 | Symbol count: 13

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
```

```
| on_read_error | 3 |
| frame_to_record | 3 |
| append_record | 3 |
| new | 2 |
| on_rx_can | 2 |
| on_rx_serial | 2 |
| on_write_error | 2 |
| on_tx_blocked | 2 |
| on_tx_can_allowed | 2 |
| on_tx_serial_allowed | 2 |
| default | 1 |
| inject_rx | 1 |
| inject_disconnect | 1 |
| clear_disconnect | 1 |
| init | 1 |
| from_path | 1 |
| read_frame | 1 |
| try_read_frame | 1 |
| send_frame | 1 |
| write_intent_enabled | 1 |
| is_allowed | 1 |
| check_can | 1 |
| check_serial | 1 |
| on_rate_limited | 1 |
| close | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 11 | struct | MockAdapter | data/schema coupling and match/serde break risk |
| 23 | impl | MockAdapter | method behavior and trait semantics |
| 24 | fn | new | API and behavior coupling to callers/implementers |
| 28 | fn | inject_rx | API and behavior coupling to callers/implementers |
| 32 | fn | inject_disconnect | API and behavior coupling to callers/implementers |
| 36 | fn | clear_disconnect | API and behavior coupling to callers/implementers |
| 41 | impl | HardwareInterface for MockAdapter | method behavior and trait semantics |
| 42 | fn | init | API and behavior coupling to callers/implementers |
| 61 | fn | read_frame | API and behavior coupling to callers/implementers |
| 89 | fn | try_read_frame | API and behavior coupling to callers/implementers |
| 114 | fn | send_frame | API and behavior coupling to callers/implementers |
| 176 | fn | close | API and behavior coupling to callers/implementers |
| 182 | fn | now_ms | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::path::PathBuf; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use super::allowlist::Allowlist; | import wiring; changing path/name can break compilation and module linkage. |
| 5 | use super::hw::{Frame, HardwareInterface, HwConfig, HwError}; | import wiring; changing path/name can break compil
| 6 | use super::metrics::HwMetrics; | import wiring; changing path/name can break compilation and module linkage. |
| 7 | use super::rate_limiter::TwoLevelRateLimiter; | import wiring; changing path/name can break compilation and module
| 8 | use super::replay::{append_record, frame_to_record}; | import wiring; changing path/name can break compilation and
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | #[derive(Debug, Default)] | comment/documentation; changing affects understanding and intent traceability. |
| 11 | pub struct MockAdapter { | data layout contract; field changes can break callers/serde/tests. |
| 12 |     config: Option<HwConfig>, | support logic; impact depends on neighboring block and data flow. |
| 13 |     closed: bool, | support logic; impact depends on neighboring block and data flow. |
| 14 |     disconnected: bool, | support logic; impact depends on neighboring block and data flow. |
| 15 |     pub rx_queue: VecDeque<Frame>, | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub tx_queue: VecDeque<Frame>, | support logic; impact depends on neighboring block and data flow. |
| 17 |     allowlist: Option<Allowlist>, | support logic; impact depends on neighboring block and data flow. |
| 18 |     limiter: Option<TwoLevelRateLimiter>, | support logic; impact depends on neighboring block and data flow. |
| 19 |     log_path: Option<PathBuf>, | support logic; impact depends on neighboring block and data flow. |
| 20 |     pub metrics: HwMetrics, | support logic; impact depends on neighboring block and data flow. |
| 21 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 22 | (blank) | blank separator; no runtime behavior. |
| 23 | impl MockAdapter { | implementation binding; behavior semantics and trait conformance live here. |
| 24 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 25 |         Self::default() | support logic; impact depends on neighboring block and data flow. |
| 26 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 | (blank) | blank separator; no runtime behavior. |
```

```
| 28 |     pub fn inject_rx(&mut self, frame: Frame) { | function signature/body; changes affect API behavior and contro
| 29 |         self.rx_queue.push_back(frame); | support logic; impact depends on neighboring block and data flow. |
| 30 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 |     pub fn inject_disconnect(&mut self) { | function signature/body; changes affect API behavior and control-flow
| 33 |         self.disconnected = true; | support logic; impact depends on neighboring block and data flow. |
| 34 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 35 | (blank) | blank separator; no runtime behavior. |
| 36 |     pub fn clear_disconnect(&mut self) { | function signature/body; changes affect API behavior and control-flow
| 37 |         self.disconnected = false; | support logic; impact depends on neighboring block and data flow. |
| 38 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 40 | (blank) | blank separator; no runtime behavior. |
| 41 | impl HardwareInterface for MockAdapter { | implementation binding; behavior semantics and trait conformance live
| 42 |     fn init(&mut self, config: &HwConfig) -> Result<(), HwError> { | function signature/body; changes affect API
| 43 |         self.config = Some(config.clone()); | support logic; impact depends on neighboring block and data flow.
| 44 |         self.closed = false; | support logic; impact depends on neighboring block and data flow. |
| 45 |         self.allowlist = if let Some(path) = &config.allowlist_path { | support logic; impact depends on neighbo
| 46 |             Some( | support logic; impact depends on neighboring block and data flow. |
| 47 |                 Allowlist::from_path(path) | support logic; impact depends on neighboring block and data flow. |
| 48 |                     .map_err(\|e\| HwError::Unknown(format!("load allowlist failed: {e}")))?, | error propagatio
| 49 |             ) | support logic; impact depends on neighboring block and data flow. |
| 50 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 51 |             None | support logic; impact depends on neighboring block and data flow. |
| 52 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 53 |         self.limiter = Some(TwoLevelRateLimiter::new( | support logic; impact depends on neighboring block and da
| 54 |             config.rate_limit_global_per_sec, | support logic; impact depends on neighboring block and data flow
| 55 |             config.rate_limit_per_id_per_sec, | support logic; impact depends on neighboring block and data flow
| 56 |         )); | support logic; impact depends on neighboring block and data flow. |
| 57 |         self.log_path = Some(config.log_dir.join("mock-session.jsonl")); | support logic; impact depends on neigh
| 58 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 59 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 |     fn read_frame(&mut self) -> Result<Frame, HwError> { | function signature/body; changes affect API behavior a
| 62 |         if self.disconnected { | runtime gate; condition changes can enable/disable critical paths. |
| 63 |             self.metrics.on_read_error(); | support logic; impact depends on neighboring block and data flow. |
| 64 |             return Err(HwError::Timeout); | support logic; impact depends on neighboring block and data flow. |
| 65 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 66 | (blank) | blank separator; no runtime behavior. |
| 67 |         let frame = self.rx_queue.pop_front().ok_or_else(\|\| { | support logic; impact depends on neighboring b
| 68 |             self.metrics.on_read_error(); | support logic; impact depends on neighboring block and data flow. |
| 69 |             HwError::Timeout | support logic; impact depends on neighboring block and data flow. |
| 70 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 71 | (blank) | blank separator; no runtime behavior. |
| 72 |         match &frame { | branch dispatcher; arm changes alter behavior class handling. |
| 73 |             Frame::Can(cf) => self | support logic; impact depends on neighboring block and data flow. |
| 74 |                 .metrics | support logic; impact depends on neighboring block and data flow. |
| 75 |                 .on_rx_can(cf.timestamp_ms.unwrap_or_else(now_ms)), | support logic; impact depends on neighborin
| 76 |             Frame::Serial(sf) => self | support logic; impact depends on neighboring block and data flow. |
| 77 |                 .metrics | support logic; impact depends on neighboring block and data flow. |
| 78 |                 .on_rx_serial(sf.timestamp_ms.unwrap_or_else(now_ms)), | support logic; impact depends on neighbo
| 79 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 80 | (blank) | blank separator; no runtime behavior. |
| 81 |         if let Some(path) = &self.log_path { | runtime gate; condition changes can enable/disable critical paths
| 82 |             let rec = frame_to_record(&frame, "rx", None, None); | support logic; impact depends on neighboring b
| 83 |             let _ = append_record(path, &rec); | support logic; impact depends on neighboring block and data flow
| 84 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 85 | (blank) | blank separator; no runtime behavior. |
| 86 |         Ok(frame) | support logic; impact depends on neighboring block and data flow. |
| 87 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 88 | (blank) | blank separator; no runtime behavior. |
| 89 |     fn try_read_frame(&mut self) -> Result<Option<Frame>, HwError> { | function signature/body; changes affect AP
| 90 |         if self.disconnected { | runtime gate; condition changes can enable/disable critical paths. |
| 91 |             self.metrics.on_read_error(); | support logic; impact depends on neighboring block and data flow. |
| 92 |             return Err(HwError::Timeout); | support logic; impact depends on neighboring block and data flow. |
| 93 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 94 | (blank) | blank separator; no runtime behavior. |
| 95 |         let out = self.rx_queue.pop_front(); | support logic; impact depends on neighboring block and data flow.
| 96 |         if let Some(frame) = &out { | runtime gate; condition changes can enable/disable critical paths. |
| 97 |             match frame { | branch dispatcher; arm changes alter behavior class handling. |
| 98 |                 Frame::Can(cf) => self | support logic; impact depends on neighboring block and data flow. |
| 99 |                     .metrics | support logic; impact depends on neighboring block and data flow. |
| 100 |                     .on_rx_can(cf.timestamp_ms.unwrap_or_else(now_ms)), | support logic; impact depends on neigh
| 101 |                 Frame::Serial(sf) => self | support logic; impact depends on neighboring block and data flow. |
| 102 |                     .metrics | support logic; impact depends on neighboring block and data flow. |
| 103 |                     .on_rx_serial(sf.timestamp_ms.unwrap_or_else(now_ms)), | support logic; impact depends on ne
```

```
| 104 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 105 |             if let Some(path) = &self.log_path { | runtime gate; condition changes can enable/disable critical |
| 106 |                 let rec = frame_to_record(frame, "rx", None, None); | support logic; impact depends on neighbori
| 107 |                 let _ = append_record(path, &rec); | support logic; impact depends on neighboring block and data
| 108 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 109 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 110 | (blank) | blank separator; no runtime behavior. |
| 111 |         Ok(out) | support logic; impact depends on neighboring block and data flow. |
| 112 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 113 | (blank) | blank separator; no runtime behavior. |
| 114 |     fn send_frame(&mut self, frame: Frame) -> Result<(), HwError> { | function signature/body; changes affect AP
| 115 |         if self.disconnected { | runtime gate; condition changes can enable/disable critical paths. |
| 116 |             self.metrics.on_write_error(); | protocol or I/O critical; changes may impact external integration
| 117 |             return Err(HwError::TransceiverError); | support logic; impact depends on neighboring block and data
| 118 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 119 | (blank) | blank separator; no runtime behavior. |
| 120 |         let cfg = self | support logic; impact depends on neighboring block and data flow. |
| 121 |             .config | support logic; impact depends on neighboring block and data flow. |
| 122 |             .as_ref() | support logic; impact depends on neighboring block and data flow. |
| 123 |             .ok_or_else(\|\| HwError::Unknown("adapter not initialized".to_string()))?; | error propagation poi
| 124 | (blank) | blank separator; no runtime behavior. |
| 125 |         if !cfg.write_intent_enabled() { | runtime gate; condition changes can enable/disable critical paths. |
| 126 |             self.metrics.on_tx_blocked(); | support logic; impact depends on neighboring block and data flow. |
| 127 |             return Err(HwError::WriteBlockedAllowlist); | support logic; impact depends on neighboring block an
| 128 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 129 | (blank) | blank separator; no runtime behavior. |
| 130 |         let is_allowed = self | support logic; impact depends on neighboring block and data flow. |
| 131 |             .allowlist | support logic; impact depends on neighboring block and data flow. |
| 132 |             .as_ref() | support logic; impact depends on neighboring block and data flow. |
| 133 |             .is_some_and(\|al\| al.is_allowed(&frame)); | support logic; impact depends on neighboring block an
| 134 |         if !is_allowed { | runtime gate; condition changes can enable/disable critical paths. |
| 135 |             self.metrics.on_tx_blocked(); | support logic; impact depends on neighboring block and data flow. |
| 136 |             return Err(HwError::WriteBlockedAllowlist); | support logic; impact depends on neighboring block an
| 137 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 | (blank) | blank separator; no runtime behavior. |
| 139 |         let is_rate_ok = if let Some(lim) = self.limiter.as_mut() { | support logic; impact depends on neighbori
| 140 |             match &frame { | branch dispatcher; arm changes alter behavior class handling. |
| 141 |                 Frame::Can(cf) => lim.check_can(cf.id), | support logic; impact depends on neighboring block an
| 142 |                 Frame::Serial(_) => lim.check_serial(), | support logic; impact depends on neighboring block an
| 143 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 144 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 145 |             true | support logic; impact depends on neighboring block and data flow. |
| 146 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 147 | (blank) | blank separator; no runtime behavior. |
| 148 |         if !is_rate_ok { | runtime gate; condition changes can enable/disable critical paths. |
| 149 |             self.metrics.on_rate_limited(); | support logic; impact depends on neighboring block and data flow.
| 150 |             self.metrics.on_write_error(); | protocol or I/O critical; changes may impact external integration
| 151 |             return Err(HwError::RateLimited); | support logic; impact depends on neighboring block and data flo
| 152 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 153 | (blank) | blank separator; no runtime behavior. |
| 154 |         if let Some(path) = &self.log_path { | runtime gate; condition changes can enable/disable critical path
| 155 |             let rec = frame_to_record(&frame, "tx", Some(true), Some(cfg.dry_run)); | support logic; impact dep
| 156 |             let _ = append_record(path, &rec); | support logic; impact depends on neighboring block and data fl
| 157 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 158 | (blank) | blank separator; no runtime behavior. |
| 159 |         if cfg.dry_run { | runtime gate; condition changes can enable/disable critical paths. |
| 160 |             match &frame { | branch dispatcher; arm changes alter behavior class handling. |
| 161 |                 Frame::Can(_) => self.metrics.on_tx_can_allowed(), | support logic; impact depends on neighbori
| 162 |                 Frame::Serial(_) => self.metrics.on_tx_serial_allowed(), | support logic; impact depends on nei
| 163 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 164 |             return Ok(()); | support logic; impact depends on neighboring block and data flow. |
| 165 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 | (blank) | blank separator; no runtime behavior. |
| 167 |         self.tx_queue.push_back(frame.clone()); | support logic; impact depends on neighboring block and data f
| 168 |         match frame { | branch dispatcher; arm changes alter behavior class handling. |
| 169 |             Frame::Can(_) => self.metrics.on_tx_can_allowed(), | support logic; impact depends on neighboring b
| 170 |             Frame::Serial(_) => self.metrics.on_tx_serial_allowed(), | support logic; impact depends on neighbo
| 171 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 172 | (blank) | blank separator; no runtime behavior. |
| 173 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 174 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 175 | (blank) | blank separator; no runtime behavior. |
| 176 |     fn close(&mut self) -> Result<(), HwError> { | function signature/body; changes affect API behavior and con
| 177 |         self.closed = true; | support logic; impact depends on neighboring block and data flow. |
| 178 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 179 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 180 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 181 | (blank) | blank separator; no runtime behavior. |
| 182 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 183 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 184 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data flo
| 185 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 186 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 187 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/io/mod.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 15 | Non-empty: 15 | Symbol count: 13

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| all | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 1 | module | allowlist | namespace and compile-time linkage |
| 2 | module | artifact | namespace and compile-time linkage |
| 3 | module | ecm_params | namespace and compile-time linkage |
| 4 | module | hw | namespace and compile-time linkage |
| 5 | module | live_runner | namespace and compile-time linkage |
| 6 | module | metrics | namespace and compile-time linkage |
| 7 | module | mock | namespace and compile-time linkage |
| 9 | module | production_program | namespace and compile-time linkage |
| 10 | module | rate_limiter | namespace and compile-time linkage |
| 11 | module | replay | namespace and compile-time linkage |
| 12 | module | serial_adapter | namespace and compile-time linkage |
| 13 | module | socketcan_adapter | namespace and compile-time linkage |
| 15 | module | vendor_cat_comm | namespace and compile-time linkage |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | pub mod allowlist; | module declaration; changing can break namespace graph. |
| 2 | pub mod artifact; | module declaration; changing can break namespace graph. |
| 3 | pub mod ecm_params; | module declaration; changing can break namespace graph. |
| 4 | pub mod hw; | module declaration; changing can break namespace graph. |
| 5 | pub mod live_runner; | module declaration; changing can break namespace graph. |
| 6 | pub mod metrics; | module declaration; changing can break namespace graph. |
| 7 | pub mod mock; | module declaration; changing can break namespace graph. |
| 8 | #[cfg(not(test))] | comment/documentation; changing affects understanding and intent traceability. |
| 9 | pub mod production_program; | module declaration; changing can break namespace graph. |
| 10 | pub mod rate_limiter; | module declaration; changing can break namespace graph. |
| 11 | pub mod replay; | module declaration; changing can break namespace graph. |
| 12 | pub mod serial_adapter; | module declaration; changing can break namespace graph. |
| 13 | pub mod socketcan_adapter; | module declaration; changing can break namespace graph. |
| 14 | #[cfg(all(target_os = "windows", feature = "vendor-windows"))] | comment/documentation; changing affects understa
| 15 | pub mod vendor_cat_comm; | module declaration; changing can break namespace graph. |
```

# File Deep Dive: src/io/production_program.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 797 | Non-empty: 750 | Symbol count: 18

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| push | 24 |
| new | 15 |
```

```
| key_advance | 12 |
| len | 6 |
| process | 5 |
| tick | 4 |
| missing_for_live_flash | 3 |
| is_empty | 3 |
| write_effectively_enabled | 3 |
| run_sim_stress_check | 3 |
| now_ms | 3 |
| run_sim_baseline_check | 2 |
| capabilities_summary | 2 |
| run_sim_connectivity_check | 2 |
| run_sim_identity_check | 2 |
| run_sim_preflight_check | 2 |
| run_uds_runtime_self_test | 2 |
| all | 2 |
| build_conformance_summary | 2 |
| hex_sha | 2 |
| default | 1 |
| run_all_phases | 1 |
| declared_capabilities | 1 |
| detect_ecms | 1 |
| connect_ecm | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 18 | enum | PhaseStatus | data/schema coupling and match/serde break risk |
| 25 | struct | PhaseResult | data/schema coupling and match/serde break risk |
| 33 | struct | ProgramReport | data/schema coupling and match/serde break risk |
| 50 | struct | ServiceConformanceResult | data/schema coupling and match/serde break risk |
| 57 | struct | ConformanceSummary | data/schema coupling and match/serde break risk |
| 63 | struct | ProgramOptions | data/schema coupling and match/serde break risk |
| 68 | impl | Default for ProgramOptions | method behavior and trait semantics |
| 69 | fn | default | API and behavior coupling to callers/implementers |
| 77 | fn | run_all_phases | API and behavior coupling to callers/implementers |
| 568 | fn | run_sim_baseline_check | API and behavior coupling to callers/implementers |
| 587 | fn | run_sim_connectivity_check | API and behavior coupling to callers/implementers |
| 601 | fn | run_sim_identity_check | API and behavior coupling to callers/implementers |
| 613 | fn | build_conformance_summary | API and behavior coupling to callers/implementers |
| 731 | fn | run_sim_preflight_check | API and behavior coupling to callers/implementers |
| 744 | fn | run_uds_runtime_self_test | API and behavior coupling to callers/implementers |
| 763 | fn | run_sim_stress_check | API and behavior coupling to callers/implementers |
| 784 | fn | hex_sha | API and behavior coupling to callers/implementers |
| 792 | fn | now_ms | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::fs; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::path::{Path, PathBuf}; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | use std::time::{Duration, SystemTime, UNIX_EPOCH}; | import wiring; changing path/name can break compilation and m
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 6 | use sha2::{Digest, Sha256}; | import wiring; changing path/name can break compilation and module linkage. |
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | use crate::HeavyMachinery; | import wiring; changing path/name can break compilation and module linkage. |
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | use super::artifact::{validate_firmware_artifact, FirmwareArtifactReport}; | import wiring; changing path/name ca
| 11 | use super::hw::{HwCapabilities, HwConfig, HwError, HwMode}; | import wiring; changing path/name can break compila
| 12 | use super::live_runner::{ | import wiring; changing path/name can break compilation and module linkage. |
| 13 |     connect_ecm, detect_ecms, live_flash_ecm_firmware, live_preflight_ecm, live_read_ecm_identity, | protocol or
| 14 |     DetectResult, FlashPreflightReport, FlashSummary, | protocol or I/O critical; changes may impact external int
| 15 | }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 | #[derive(Debug, Clone, Copy, Serialize, Deserialize, PartialEq, Eq)] | comment/documentation; changing affects un
| 18 | pub enum PhaseStatus { | state machine variants; changes ripple through matches and business logic. |
| 19 |     Passed, | support logic; impact depends on neighboring block and data flow. |
| 20 |     Failed, | support logic; impact depends on neighboring block and data flow. |
| 21 |     Blocked, | support logic; impact depends on neighboring block and data flow. |
| 22 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
```

```
| 25 | pub struct PhaseResult { | data layout contract; field changes can break callers/serde/tests. |
| 26 |     pub phase: u8, | support logic; impact depends on neighboring block and data flow. |
| 27 |     pub title: String, | support logic; impact depends on neighboring block and data flow. |
| 28 |     pub status: PhaseStatus, | support logic; impact depends on neighboring block and data flow. |
| 29 |     pub details: String, | support logic; impact depends on neighboring block and data flow. |
| 30 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
| 33 | pub struct ProgramReport { | data layout contract; field changes can break callers/serde/tests. |
| 34 |     pub generated_at_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 35 |     pub mode: String, | support logic; impact depends on neighboring block and data flow. |
| 36 |     pub target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 37 |     pub strict_mode: bool, | support logic; impact depends on neighboring block and data flow. |
| 38 |     pub execute_flash: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 39 |     pub artifact: Option<FirmwareArtifactReport>, | support logic; impact depends on neighboring block and data f
| 40 |     pub detect: Option<DetectResult>, | support logic; impact depends on neighboring block and data flow. |
| 41 |     pub preflight: Option<FlashPreflightReport>, | protocol or I/O critical; changes may impact external integrat
| 42 |     pub identity: Option<super::live_runner::EcuIdentity>, | support logic; impact depends on neighboring block a
| 43 |     pub conformance_summary: Option<ConformanceSummary>, | support logic; impact depends on neighboring block and
| 44 |     pub phases: Vec<PhaseResult>, | support logic; impact depends on neighboring block and data flow. |
| 45 |     pub overall_passed: bool, | support logic; impact depends on neighboring block and data flow. |
| 46 |     pub report_sha256: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 47 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 | (blank) | blank separator; no runtime behavior. |
| 49 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
| 50 | pub struct ServiceConformanceResult { | data layout contract; field changes can break callers/serde/tests. |
| 51 |     pub service_sid: String, | support logic; impact depends on neighboring block and data flow. |
| 52 |     pub passed: bool, | support logic; impact depends on neighboring block and data flow. |
| 53 |     pub evidence: String, | support logic; impact depends on neighboring block and data flow. |
| 54 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 | (blank) | blank separator; no runtime behavior. |
| 56 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
| 57 | pub struct ConformanceSummary { | data layout contract; field changes can break callers/serde/tests. |
| 58 |     pub passed: bool, | support logic; impact depends on neighboring block and data flow. |
| 59 |     pub services: Vec<ServiceConformanceResult>, | support logic; impact depends on neighboring block and data f
| 60 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 61 | (blank) | blank separator; no runtime behavior. |
| 62 | #[derive(Debug, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceability. |
| 63 | pub struct ProgramOptions { | data layout contract; field changes can break callers/serde/tests. |
| 64 |     pub strict_mode: bool, | support logic; impact depends on neighboring block and data flow. |
| 65 |     pub execute_flash: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 66 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 67 | (blank) | blank separator; no runtime behavior. |
| 68 | impl Default for ProgramOptions { | implementation binding; behavior semantics and trait conformance live here.
| 69 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 70 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 71 |             strict_mode: true, | support logic; impact depends on neighboring block and data flow. |
| 72 |             execute_flash: false, | protocol or I/O critical; changes may impact external integration correctness
| 73 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 74 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 75 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 76 | (blank) | blank separator; no runtime behavior. |
| 77 | pub fn run_all_phases( | function signature/body; changes affect API behavior and control-flow. |
| 78 |     cfg: &HwConfig, | support logic; impact depends on neighboring block and data flow. |
| 79 |     target_sa: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 80 |     artifact_path: Option<&Path>, | support logic; impact depends on neighboring block and data flow. |
| 81 |     out_dir: &Path, | support logic; impact depends on neighboring block and data flow. |
| 82 |     options: ProgramOptions, | support logic; impact depends on neighboring block and data flow. |
| 83 | ) -> Result<PathBuf, HwError> { | support logic; impact depends on neighboring block and data flow. |
| 84 |     fs::create_dir_all(out_dir) | support logic; impact depends on neighboring block and data flow. |
| 85 |         .map_err(\|e\| HwError::Unknown(format!("create report dir failed: {e}")))?; | error propagation point; c
| 86 | (blank) | blank separator; no runtime behavior. |
| 87 |     let mut phases = Vec::with_capacity(12); | support logic; impact depends on neighboring block and data flow.
| 88 |     let mut artifact = None; | support logic; impact depends on neighboring block and data flow. |
| 89 |     let mut detect = None; | support logic; impact depends on neighboring block and data flow. |
| 90 |     let mut preflight = None; | support logic; impact depends on neighboring block and data flow. |
| 91 |     let mut identity = None; | support logic; impact depends on neighboring block and data flow. |
| 92 |     let mut flash_summary: Option<FlashSummary> = None; | protocol or I/O critical; changes may impact external i
| 93 |     let capabilities = cfg.declared_capabilities(); | support logic; impact depends on neighboring block and data
| 94 |     let missing_live_caps = capabilities.missing_for_live_flash(); | protocol or I/O critical; changes may impact
| 95 | (blank) | blank separator; no runtime behavior. |
| 96 |     // Phase 1: Compliance baseline | comment/documentation; changing affects understanding and intent traceabili
| 97 |     let docs_ok = Path::new("docs/ECM-Data.md").exists() | support logic; impact depends on neighboring block and
| 98 |         && Path::new("docs/CAT_COMM_TEMPLATE_PROTOCOL.md").exists(); | support logic; impact depends on neighbori
| 99 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 100 |         phase: 1, | support logic; impact depends on neighboring block and data flow. |
```

```
| 101 |         title: "Baseline e Compliance".to_string(), | support logic; impact depends on neighboring block and da
| 102 |         status: if docs_ok { | support logic; impact depends on neighboring block and data flow. |
| 103 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 104 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 105 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 106 |         }, | support logic; impact depends on neighboring block and data flow. |
| 107 |         details: if docs_ok { | support logic; impact depends on neighboring block and data flow. |
| 108 |             "runbook e protocolo presentes".to_string() | support logic; impact depends on neighboring block and
| 109 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 110 |             "faltam documentos obrigatorios em docs/".to_string() | support logic; impact depends on neighboring
| 111 |         }, | support logic; impact depends on neighboring block and data flow. |
| 112 |     }); | support logic; impact depends on neighboring block and data flow. |
| 113 | (blank) | blank separator; no runtime behavior. |
| 114 |     // Phase 2: Hardware bench readiness | comment/documentation; changing affects understanding and intent tra
| 115 |     let hw_ready = matches!(cfg.mode, HwMode::Live) && missing_live_caps.is_empty(); | support logic; impact de
| 116 |     let sim_baseline_ok = run_sim_baseline_check(); | support logic; impact depends on neighboring block and da
| 117 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 118 |         phase: 2, | support logic; impact depends on neighboring block and data flow. |
| 119 |         title: "Plataforma Fisica Real".to_string(), | support logic; impact depends on neighboring block and d
| 120 |         status: if hw_ready { | support logic; impact depends on neighboring block and data flow. |
| 121 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 122 |         } else if matches!(cfg.mode, HwMode::Sim) && sim_baseline_ok.0 { | support logic; impact depends on nei
| 123 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 124 |         } else if matches!(cfg.mode, HwMode::Sim) { | support logic; impact depends on neighboring block and da
| 125 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 126 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 127 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 128 |         }, | support logic; impact depends on neighboring block and data flow. |
| 129 |         details: if hw_ready { | support logic; impact depends on neighboring block and data flow. |
| 130 |             format!("capabilities: {}", cfg.capabilities_summary()) | support logic; impact depends on neighbor
| 131 |         } else if matches!(cfg.mode, HwMode::Sim) { | support logic; impact depends on neighboring block and da
| 132 |             format!("sim_harness={}: {}", sim_baseline_ok.0, sim_baseline_ok.1) | support logic; impact depends
| 133 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 134 |             format!( | support logic; impact depends on neighboring block and data flow. |
| 135 |                 "capabilities insufficients: {}; missing={}", | support logic; impact depends on neighboring bl
| 136 |                 cfg.capabilities_summary(), | support logic; impact depends on neighboring block and data flow.
| 137 |                 if missing_live_caps.is_empty() { | runtime gate; condition changes can enable/disable critical
| 138 |                     "none".to_string() | support logic; impact depends on neighboring block and data flow. |
| 139 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 140 |                     missing_live_caps.join(",") | support logic; impact depends on neighboring block and data f
| 141 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 142 |             ) | support logic; impact depends on neighboring block and data flow. |
| 143 |         }, | support logic; impact depends on neighboring block and data flow. |
| 144 |     }); | support logic; impact depends on neighboring block and data flow. |
| 145 | (blank) | blank separator; no runtime behavior. |
| 146 |     // Phase 3: Live detect/connect | comment/documentation; changing affects understanding and intent traceabi
| 147 |     if hw_ready { | runtime gate; condition changes can enable/disable critical paths. |
| 148 |         match detect_ecms(cfg, Duration::from_secs(3)) { | branch dispatcher; arm changes alter behavior class
| 149 |             Ok(d) => { | support logic; impact depends on neighboring block and data flow. |
| 150 |                 let found = d.source_addresses.len(); | support logic; impact depends on neighboring block and
| 151 |                 detect = Some(d); | support logic; impact depends on neighboring block and data flow. |
| 152 |                 phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 153 |                     phase: 3, | support logic; impact depends on neighboring block and data flow. |
| 154 |                     title: "Conexao Confiavel".to_string(), | support logic; impact depends on neighboring bloc
| 155 |                     status: if found > 0 { | support logic; impact depends on neighboring block and data flow.
| 156 |                         PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow.
| 157 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 158 |                         PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow.
| 159 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 160 |                     details: format!("ECMs detectados: {found}"), | support logic; impact depends on neighborin
| 161 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 162 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 163 |             Err(e) => phases.push(PhaseResult { | support logic; impact depends on neighboring block and data f
| 164 |                 phase: 3, | support logic; impact depends on neighboring block and data flow. |
| 165 |                 title: "Conexao Confiavel".to_string(), | support logic; impact depends on neighboring block an
| 166 |                 status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow
| 167 |                 details: format!("falha detect/connect: {e}"), | support logic; impact depends on neighboring b
| 168 |             }), | support logic; impact depends on neighboring block and data flow. |
| 169 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 170 | (blank) | blank separator; no runtime behavior. |
| 171 |         match connect_ecm(cfg, target_sa) { | branch dispatcher; arm changes alter behavior class handling. |
| 172 |             Ok(()) => {} | support logic; impact depends on neighboring block and data flow. |
| 173 |             Err(e) => phases.push(PhaseResult { | support logic; impact depends on neighboring block and data f
| 174 |                 phase: 3, | support logic; impact depends on neighboring block and data flow. |
| 175 |                 title: "Conexao Confiavel (Connect)".to_string(), | support logic; impact depends on neighborin
| 176 |                 status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow
```

```
| 177 |                 details: format!("connect falhou: {e}"), | support logic; impact depends on neighboring block ar
| 178 |             }), | support logic; impact depends on neighboring block and data flow. |
| 179 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 180 |     } else if matches!(cfg.mode, HwMode::Sim) { | support logic; impact depends on neighboring block and data fl
| 181 |         let (ok, detail) = run_sim_connectivity_check(); | support logic; impact depends on neighboring block ar
| 182 |         phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 183 |             phase: 3, | support logic; impact depends on neighboring block and data flow. |
| 184 |             title: "Conexao Confiavel".to_string(), | support logic; impact depends on neighboring block and dat
| 185 |             status: if ok { PhaseStatus::Passed } else { PhaseStatus::Failed }, | support logic; impact depends
| 186 |             details: detail, | support logic; impact depends on neighboring block and data flow. |
| 187 |         }); | support logic; impact depends on neighboring block and data flow. |
| 188 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 189 |         phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 190 |             phase: 3, | support logic; impact depends on neighboring block and data flow. |
| 191 |             title: "Conexao Confiavel".to_string(), | support logic; impact depends on neighboring block and dat
| 192 |             status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow. |
| 193 |             details: "hardware live indisponivel".to_string(), | support logic; impact depends on neighboring bl
| 194 |         }); | support logic; impact depends on neighboring block and data flow. |
| 195 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 196 | (blank) | blank separator; no runtime behavior. |
| 197 |     // Phase 4: ECU identity | comment/documentation; changing affects understanding and intent traceability. |
| 198 |     if hw_ready { | runtime gate; condition changes can enable/disable critical paths. |
| 199 |         match live_read_ecm_identity(cfg, target_sa) { | branch dispatcher; arm changes alter behavior class han
| 200 |             Ok(id) => { | support logic; impact depends on neighboring block and data flow. |
| 201 |                 let fp = id.fingerprint.is_some(); | support logic; impact depends on neighboring block and data
| 202 |                 identity = Some(id); | support logic; impact depends on neighboring block and data flow. |
| 203 |                 phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 204 |                     phase: 4, | support logic; impact depends on neighboring block and data flow. |
| 205 |                     title: "Leitura e Identidade ECM".to_string(), | support logic; impact depends on neighborin
| 206 |                     status: if fp \|\| cfg.allow_untrusted_ecu { | support logic; impact depends on neighboring
| 207 |                         PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow.
| 208 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 209 |                         PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow.
| 210 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 211 |                     details: format!("fingerprint={fp} allow_untrusted_ecu={}", cfg.allow_untrusted_ecu), | supp
| 212 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 213 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 214 |             Err(e) => phases.push(PhaseResult { | support logic; impact depends on neighboring block and data fl
| 215 |                 phase: 4, | support logic; impact depends on neighboring block and data flow. |
| 216 |                 title: "Leitura e Identidade ECM".to_string(), | support logic; impact depends on neighboring bl
| 217 |                 status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow
| 218 |                 details: format!("falha leitura DID: {e}"), | support logic; impact depends on neighboring bloc
| 219 |             }), | support logic; impact depends on neighboring block and data flow. |
| 220 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 221 |     } else if matches!(cfg.mode, HwMode::Sim) { | support logic; impact depends on neighboring block and data fl
| 222 |         let (ok, detail) = run_sim_identity_check(); | support logic; impact depends on neighboring block and da
| 223 |         phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 224 |             phase: 4, | support logic; impact depends on neighboring block and data flow. |
| 225 |             title: "Leitura e Identidade ECM".to_string(), | support logic; impact depends on neighboring block
| 226 |             status: if ok { PhaseStatus::Passed } else { PhaseStatus::Failed }, | support logic; impact depends
| 227 |             details: detail, | support logic; impact depends on neighboring block and data flow. |
| 228 |         }); | support logic; impact depends on neighboring block and data flow. |
| 229 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 230 |         phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 231 |             phase: 4, | support logic; impact depends on neighboring block and data flow. |
| 232 |             title: "Leitura e Identidade ECM".to_string(), | support logic; impact depends on neighboring block
| 233 |             status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow. |
| 234 |             details: "requer modo live".to_string(), | support logic; impact depends on neighboring block and da
| 235 |         }); | support logic; impact depends on neighboring block and data flow. |
| 236 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 237 | (blank) | blank separator; no runtime behavior. |
| 238 |     // Phase 5: Artifact validation | comment/documentation; changing affects understanding and intent traceabil
| 239 |     match artifact_path { | branch dispatcher; arm changes alter behavior class handling. |
| 240 |         Some(path) => match validate_firmware_artifact(path, 64 * 1024 * 1024) { | support logic; impact depends
| 241 |             Ok(r) => { | support logic; impact depends on neighboring block and data flow. |
| 242 |                 artifact = Some(r.clone()); | support logic; impact depends on neighboring block and data flow.
| 243 |                 phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 244 |                     phase: 5, | support logic; impact depends on neighboring block and data flow. |
| 245 |                     title: "Validacao de Artefato".to_string(), | support logic; impact depends on neighboring b
| 246 |                     status: PhaseStatus::Passed, | support logic; impact depends on neighboring block and data f
| 247 |                     details: format!("{} bytes crc32=0x{:08X}", r.bytes, r.crc32), | support logic; impact depen
| 248 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 249 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 250 |             Err(e) => phases.push(PhaseResult { | support logic; impact depends on neighboring block and data fl
| 251 |                 phase: 5, | support logic; impact depends on neighboring block and data flow. |
| 252 |                 title: "Validacao de Artefato".to_string(), | support logic; impact depends on neighboring bloc
```

```
| 253 |                 status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow
| 254 |                 details: format!("artefato invalido: {e}"), | support logic; impact depends on neighboring bloc
| 255 |             }), | support logic; impact depends on neighboring block and data flow. |
| 256 |         }, | support logic; impact depends on neighboring block and data flow. |
| 257 |         None => phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 258 |             phase: 5, | support logic; impact depends on neighboring block and data flow. |
| 259 |             title: "Validacao de Artefato".to_string(), | support logic; impact depends on neighboring block an
| 260 |             status: if options.strict_mode { | support logic; impact depends on neighboring block and data flow
| 261 |                 PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 262 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 263 |                 PhaseStatus::Blocked | support logic; impact depends on neighboring block and data flow. |
| 264 |             }, | support logic; impact depends on neighboring block and data flow. |
| 265 |             details: "forneca --firmware=<arquivo.bin>".to_string(), | support logic; impact depends on neighbo
| 266 |         }), | support logic; impact depends on neighboring block and data flow. |
| 267 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 268 | (blank) | blank separator; no runtime behavior. |
| 269 |     // Phase 6: Flash preflight gate | comment/documentation; changing affects understanding and intent traceab
| 270 |     if hw_ready { | runtime gate; condition changes can enable/disable critical paths. |
| 271 |         match live_preflight_ecm(cfg, target_sa) { | branch dispatcher; arm changes alter behavior class handli
| 272 |             Ok(pf) => { | support logic; impact depends on neighboring block and data flow. |
| 273 |                 let v_ok = pf.supply_voltage_v >= 11.8; | support logic; impact depends on neighboring block an
| 274 |                 preflight = Some(pf.clone()); | support logic; impact depends on neighboring block and data flo
| 275 |                 let mut status = if v_ok { | support logic; impact depends on neighboring block and data flow.
| 276 |                     PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 277 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 278 |                     PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 279 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 280 |                 let mut details = format!( | support logic; impact depends on neighboring block and data flow.
| 281 |                     "supply_v={:.2} trusted_fp={}", | support logic; impact depends on neighboring block and da
| 282 |                     pf.supply_voltage_v, pf.trusted_fingerprint | support logic; impact depends on neighboring b
| 283 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 284 |                 if options.execute_flash { | runtime gate; condition changes can enable/disable critical paths.
| 285 |                     if let (true, Some(path), true) = ( | runtime gate; condition changes can enable/disable cri
| 286 |                         status == PhaseStatus::Passed, | support logic; impact depends on neighboring block and
| 287 |                         artifact_path, | support logic; impact depends on neighboring block and data flow. |
| 288 |                         cfg.write_effectively_enabled(), | protocol or I/O critical; changes may impact externa
| 289 |                     ) { | support logic; impact depends on neighboring block and data flow. |
| 290 |                         match fs::read(path) { | branch dispatcher; arm changes alter behavior class handling.
| 291 |                             Ok(payload) => match live_flash_ecm_firmware(cfg, target_sa, &payload) { | protocol
| 292 |                                 Ok(sum) => { | support logic; impact depends on neighboring block and data flow
| 293 |                                     flash_summary = Some(sum.clone()); | protocol or I/O critical; changes may i
| 294 |                                     details = format!( | support logic; impact depends on neighboring block and
| 295 |                                         "{} \| flash_ok bytes={} blocks={} crc32=0x{:08X} \| td_ack={}/{} seq_e
| 296 |                                         details, | support logic; impact depends on neighboring block and data f
| 297 |                                         sum.bytes_sent, | support logic; impact depends on neighboring block an
| 298 |                                         sum.blocks_sent, | support logic; impact depends on neighboring block a
| 299 |                                         sum.crc32, | support logic; impact depends on neighboring block and data
| 300 |                                         sum.transport_diagnostics.transfer_data_blocks_acked, | support logic; i
| 301 |                                         sum.transport_diagnostics.transfer_data_blocks_attempted, | support log
| 302 |                                         sum.transport_diagnostics.sequence_error_count, | support logic; impact
| 303 |                                         sum.transport_diagnostics.flowcontrol_timeout_count | support logic; im
| 304 |                                     ); | support logic; impact depends on neighboring block and data flow. |
| 305 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/life
| 306 |                                 Err(e) => { | support logic; impact depends on neighboring block and data flow.
| 307 |                                     status = PhaseStatus::Failed; | support logic; impact depends on neighborin
| 308 |                                     details = format!("{} \| flash_fail={}", details, e); | protocol or I/O cri
| 309 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/life
| 310 |                             }, | support logic; impact depends on neighboring block and data flow. |
| 311 |                             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 312 |                                 status = PhaseStatus::Failed; | support logic; impact depends on neighboring bl
| 313 |                                 details = format!("{} \| firmware_read_fail={}", details, e); | support logic; i
| 314 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 315 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 316 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 317 |                         status = PhaseStatus::Failed; | support logic; impact depends on neighboring block and
| 318 |                         details = format!( | support logic; impact depends on neighboring block and data flow.
| 319 |                             "{} \| flash requested but missing requirements (artifact + write_enabled + prefligh
| 320 |                             details | support logic; impact depends on neighboring block and data flow. |
| 321 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 322 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 323 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 324 |                 phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 325 |                     phase: 6, | support logic; impact depends on neighboring block and data flow. |
| 326 |                     title: "Flash Seguro (Preflight)".to_string(), | protocol or I/O critical; changes may impac
| 327 |                     status, | support logic; impact depends on neighboring block and data flow. |
| 328 |                     details, | support logic; impact depends on neighboring block and data flow. |
```

```
| 329 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 330 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 331 |             Err(e) => phases.push(PhaseResult { | support logic; impact depends on neighboring block and data f
| 332 |                 phase: 6, | support logic; impact depends on neighboring block and data flow. |
| 333 |                 title: "Flash Seguro (Preflight)".to_string(), | protocol or I/O critical; changes may impact ex
| 334 |                 status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow
| 335 |                 details: format!("preflight falhou: {e}"), | support logic; impact depends on neighboring block
| 336 |             }), | support logic; impact depends on neighboring block and data flow. |
| 337 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 338 |     } else if matches!(cfg.mode, HwMode::Sim) { | support logic; impact depends on neighboring block and data f
| 339 |         let (ok, detail) = run_sim_preflight_check(); | support logic; impact depends on neighboring block and d
| 340 |         phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 341 |             phase: 6, | support logic; impact depends on neighboring block and data flow. |
| 342 |             title: "Flash Seguro (Preflight)".to_string(), | protocol or I/O critical; changes may impact extern
| 343 |             status: if ok { PhaseStatus::Passed } else { PhaseStatus::Failed }, | support logic; impact depends
| 344 |             details: detail, | support logic; impact depends on neighboring block and data flow. |
| 345 |         }); | support logic; impact depends on neighboring block and data flow. |
| 346 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 347 |         phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 348 |             phase: 6, | support logic; impact depends on neighboring block and data flow. |
| 349 |             title: "Flash Seguro (Preflight)".to_string(), | protocol or I/O critical; changes may impact extern
| 350 |             status: PhaseStatus::Failed, | support logic; impact depends on neighboring block and data flow. |
| 351 |             details: "requer modo live".to_string(), | support logic; impact depends on neighboring block and da
| 352 |         }); | support logic; impact depends on neighboring block and data flow. |
| 353 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 354 | (blank) | blank separator; no runtime behavior. |
| 355 |     // Phase 7: Diagnostics readiness | comment/documentation; changing affects understanding and intent traceal
| 356 |     let diag_ready_cfg = cfg.uds_timeout_p2_ms > 0 | protocol or I/O critical; changes may impact external inte
| 357 |         && cfg.uds_timeout_p2star_ms >= cfg.uds_timeout_p2_ms | protocol or I/O critical; changes may impact ex
| 358 |         && cfg.uds_retry_count > 0; | protocol or I/O critical; changes may impact external integration correctn
| 359 |     let diag_runtime = run_uds_runtime_self_test(); | protocol or I/O critical; changes may impact external inte
| 360 |     let diag_ready = diag_ready_cfg && diag_runtime.0; | support logic; impact depends on neighboring block and
| 361 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 362 |         phase: 7, | support logic; impact depends on neighboring block and data flow. |
| 363 |         title: "Diagnostico e Debug".to_string(), | support logic; impact depends on neighboring block and data
| 364 |         status: if diag_ready { | support logic; impact depends on neighboring block and data flow. |
| 365 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 366 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 367 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 368 |         }, | support logic; impact depends on neighboring block and data flow. |
| 369 |         details: format!( | support logic; impact depends on neighboring block and data flow. |
| 370 |             "uds_retry={} p2={} p2*= {} runtime={}: {}", | protocol or I/O critical; changes may impact externa
| 371 |             cfg.uds_retry_count, | protocol or I/O critical; changes may impact external integration correctness
| 372 |             cfg.uds_timeout_p2_ms, | protocol or I/O critical; changes may impact external integration correctne
| 373 |             cfg.uds_timeout_p2star_ms, | protocol or I/O critical; changes may impact external integration corre
| 374 |             diag_runtime.0, | support logic; impact depends on neighboring block and data flow. |
| 375 |             diag_runtime.1 | support logic; impact depends on neighboring block and data flow. |
| 376 |         ), | support logic; impact depends on neighboring block and data flow. |
| 377 |     }); | support logic; impact depends on neighboring block and data flow. |
| 378 | (blank) | blank separator; no runtime behavior. |
| 379 |     // Phase 8: Governance/security | comment/documentation; changing affects understanding and intent traceabi
| 380 |     let gov = cfg.allowlist_path.is_some() && cfg.noninteractive_approved; | support logic; impact depends on n
| 381 |     let write_policy = if cfg.enable_write { | protocol or I/O critical; changes may impact external integratio
| 382 |         cfg.write_effectively_enabled() | protocol or I/O critical; changes may impact external integration cor
| 383 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 384 |         true | support logic; impact depends on neighboring block and data flow. |
| 385 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 386 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 387 |         phase: 8, | support logic; impact depends on neighboring block and data flow. |
| 388 |         title: "Governanca e Seguranca".to_string(), | support logic; impact depends on neighboring block and da
| 389 |         status: if gov && write_policy { | protocol or I/O critical; changes may impact external integration co
| 390 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 391 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 392 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 393 |         }, | support logic; impact depends on neighboring block and data flow. |
| 394 |         details: format!( | support logic; impact depends on neighboring block and data flow. |
| 395 |             "allowlist_present={} noninteractive_approved={} write_effective={}", | protocol or I/O critical; ch
| 396 |             cfg.allowlist_path.is_some(), | support logic; impact depends on neighboring block and data flow. |
| 397 |             cfg.noninteractive_approved, | support logic; impact depends on neighboring block and data flow. |
| 398 |             cfg.write_effectively_enabled() | protocol or I/O critical; changes may impact external integration
| 399 |         ), | support logic; impact depends on neighboring block and data flow. |
| 400 |     }); | support logic; impact depends on neighboring block and data flow. |
| 401 | (blank) | blank separator; no runtime behavior. |
| 402 |     // Phase 9: Reliability/stress readiness | comment/documentation; changing affects understanding and intent
| 403 |     let reliability_ok = cfg.rate_limit_global_per_sec > 0 | support logic; impact depends on neighboring block
| 404 |         && cfg.rate_limit_per_id_per_sec > 0 | support logic; impact depends on neighboring block and data flow
```

```
| 405 |         && cfg.reconnect_backoff_max_ms >= cfg.reconnect_backoff_base_ms | support logic; impact depends on neig
| 406 |         && run_sim_stress_check().0; | support logic; impact depends on neighboring block and data flow. |
| 407 |     let sim_stress = run_sim_stress_check(); | support logic; impact depends on neighboring block and data flow
| 408 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 409 |         phase: 9, | support logic; impact depends on neighboring block and data flow. |
| 410 |         title: "Confiabilidade e Stress".to_string(), | support logic; impact depends on neighboring block and d
| 411 |         status: if reliability_ok { | support logic; impact depends on neighboring block and data flow. |
| 412 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 413 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 414 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 415 |         }, | support logic; impact depends on neighboring block and data flow. |
| 416 |         details: format!( | support logic; impact depends on neighboring block and data flow. |
| 417 |             "rate_global={} rate_per_id={} reconnect={}..{} ms", | support logic; impact depends on neighboring
| 418 |             cfg.rate_limit_global_per_sec, | support logic; impact depends on neighboring block and data flow.
| 419 |             cfg.rate_limit_per_id_per_sec, | support logic; impact depends on neighboring block and data flow.
| 420 |             cfg.reconnect_backoff_base_ms, | support logic; impact depends on neighboring block and data flow.
| 421 |             cfg.reconnect_backoff_max_ms | support logic; impact depends on neighboring block and data flow. |
| 422 |         ) + &format!(" stress={}: {}", sim_stress.0, sim_stress.1), | support logic; impact depends on neighbor
| 423 |     }); | support logic; impact depends on neighboring block and data flow. |
| 424 | (blank) | blank separator; no runtime behavior. |
| 425 |     // Phase 10: Industrialization | comment/documentation; changing affects understanding and intent traceabil
| 426 |     let industrial_ok = Path::new("Cargo.toml").exists() | support logic; impact depends on neighboring block a
| 427 |         && Path::new(".github/workflows/integration-mock.yml").exists() | support logic; impact depends on neig
| 428 |         && Path::new("docs/ECM-Data.md").exists(); | support logic; impact depends on neighboring block and data
| 429 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 430 |         phase: 10, | support logic; impact depends on neighboring block and data flow. |
| 431 |         title: "Industrializacao".to_string(), | support logic; impact depends on neighboring block and data fl
| 432 |         status: if industrial_ok { | support logic; impact depends on neighboring block and data flow. |
| 433 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 434 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 435 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 436 |         }, | support logic; impact depends on neighboring block and data flow. |
| 437 |         details: if industrial_ok { | support logic; impact depends on neighboring block and data flow. |
| 438 |             "pipeline de build/test encontrado".to_string() | support logic; impact depends on neighboring bloc
| 439 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 440 |             "faltam artefatos de pipeline".to_string() | support logic; impact depends on neighboring block and
| 441 |         }, | support logic; impact depends on neighboring block and data flow. |
| 442 |     }); | support logic; impact depends on neighboring block and data flow. |
| 443 | (blank) | blank separator; no runtime behavior. |
| 444 |     // Phase 11: Homologation dossier generation (placeholder, finalized after writing artifacts) | comment/doc
| 445 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 446 |         phase: 11, | support logic; impact depends on neighboring block and data flow. |
| 447 |         title: "Homologacao Final".to_string(), | support logic; impact depends on neighboring block and data f
| 448 |         status: PhaseStatus::Passed, | support logic; impact depends on neighboring block and data flow. |
| 449 |         details: "gerando dossie json/markdown".to_string(), | support logic; impact depends on neighboring bloc
| 450 |     }); | support logic; impact depends on neighboring block and data flow. |
| 451 | (blank) | blank separator; no runtime behavior. |
| 452 |     // Phase 12: Production gate | comment/documentation; changing affects understanding and intent traceabilit
| 453 |     let overall_passed = phases | support logic; impact depends on neighboring block and data flow. |
| 454 |         .iter() | support logic; impact depends on neighboring block and data flow. |
| 455 |         .filter(\|p\| p.phase <= 10) | support logic; impact depends on neighboring block and data flow. |
| 456 |         .all(\|p\| match p.status { | support logic; impact depends on neighboring block and data flow. |
| 457 |             PhaseStatus::Passed => true, | support logic; impact depends on neighboring block and data flow. |
| 458 |             PhaseStatus::Blocked => !options.strict_mode, | support logic; impact depends on neighboring block a
| 459 |             PhaseStatus::Failed => false, | support logic; impact depends on neighboring block and data flow. |
| 460 |         }); | support logic; impact depends on neighboring block and data flow. |
| 461 |     phases.push(PhaseResult { | support logic; impact depends on neighboring block and data flow. |
| 462 |         phase: 12, | support logic; impact depends on neighboring block and data flow. |
| 463 |         title: "Gate de Producao".to_string(), | support logic; impact depends on neighboring block and data fl
| 464 |         status: if overall_passed { | support logic; impact depends on neighboring block and data flow. |
| 465 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 466 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 467 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 468 |         }, | support logic; impact depends on neighboring block and data flow. |
| 469 |         details: if overall_passed { | support logic; impact depends on neighboring block and data flow. |
| 470 |             "gates criticos satisfeitos para o contexto atual".to_string() | support logic; impact depends on n
| 471 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 472 |             "ha falhas bloqueantes nas fases anteriores".to_string() | support logic; impact depends on neighbo
| 473 |         }, | support logic; impact depends on neighboring block and data flow. |
| 474 |     }); | support logic; impact depends on neighboring block and data flow. |
| 475 | (blank) | blank separator; no runtime behavior. |
| 476 |     let conformance_summary = Some(build_conformance_summary( | support logic; impact depends on neighboring bl
| 477 |         options.execute_flash, | protocol or I/O critical; changes may impact external integration correctness.
| 478 |         options.strict_mode, | support logic; impact depends on neighboring block and data flow. |
| 479 |         &flash_summary, | protocol or I/O critical; changes may impact external integration correctness. |
| 480 |         &capabilities, | support logic; impact depends on neighboring block and data flow. |
```

```
| 481 |     )); | support logic; impact depends on neighboring block and data flow. |
| 482 | (blank) | blank separator; no runtime behavior. |
| 483 |     let report = ProgramReport { | support logic; impact depends on neighboring block and data flow. |
| 484 |         generated_at_ms: now_ms(), | support logic; impact depends on neighboring block and data flow. |
| 485 |         mode: match cfg.mode { | support logic; impact depends on neighboring block and data flow. |
| 486 |             HwMode::Sim => "sim".to_string(), | support logic; impact depends on neighboring block and data flo
| 487 |             HwMode::Live => "live".to_string(), | support logic; impact depends on neighboring block and data f
| 488 |         }, | support logic; impact depends on neighboring block and data flow. |
| 489 |         target_sa, | support logic; impact depends on neighboring block and data flow. |
| 490 |         strict_mode: options.strict_mode, | support logic; impact depends on neighboring block and data flow. |
| 491 |         execute_flash: options.execute_flash, | protocol or I/O critical; changes may impact external integratio
| 492 |         artifact, | support logic; impact depends on neighboring block and data flow. |
| 493 |         detect, | support logic; impact depends on neighboring block and data flow. |
| 494 |         preflight, | support logic; impact depends on neighboring block and data flow. |
| 495 |         identity, | support logic; impact depends on neighboring block and data flow. |
| 496 |         conformance_summary, | support logic; impact depends on neighboring block and data flow. |
| 497 |         phases, | support logic; impact depends on neighboring block and data flow. |
| 498 |         overall_passed, | support logic; impact depends on neighboring block and data flow. |
| 499 |         report_sha256: None, | support logic; impact depends on neighboring block and data flow. |
| 500 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 501 | (blank) | blank separator; no runtime behavior. |
| 502 |     let ts = now_ms(); | support logic; impact depends on neighboring block and data flow. |
| 503 |     let json_path = out_dir.join(format!("production_phase_report_{ts}.json")); | support logic; impact depends
| 504 |     let md_path = out_dir.join(format!("production_phase_report_{ts}.md")); | support logic; impact depends on
| 505 | (blank) | blank separator; no runtime behavior. |
| 506 |     let json = serde_json::to_string_pretty(&report) | support logic; impact depends on neighboring block and da
| 507 |         .map_err(\|e\| HwError::Unknown(format!("serialize report json failed: {e}")))?; | error propagation poi
| 508 |     fs::write(&json_path, json) | protocol or I/O critical; changes may impact external integration correctness
| 509 |         .map_err(\|e\| HwError::Unknown(format!("write report json failed: {e}")))?; | error propagation point;
| 510 | (blank) | blank separator; no runtime behavior. |
| 511 |     let mut md = String::new(); | support logic; impact depends on neighboring block and data flow. |
| 512 |     md.push_str("# Production Program Report\n\n"); | support logic; impact depends on neighboring block and dat
| 513 |     md.push_str(&format!("- generated_at_ms: {}\n", report.generated_at_ms)); | support logic; impact depends on
| 514 |     md.push_str(&format!("- mode: {}\n", report.mode)); | support logic; impact depends on neighboring block and
| 515 |     md.push_str(&format!("- target_sa: {:?}\n", report.target_sa)); | error propagation point; changes alter fai
| 516 |     md.push_str(&format!("- overall_passed: {}\n\n", report.overall_passed)); | support logic; impact depends on
| 517 |     if let Some(c) = &report.conformance_summary { | runtime gate; condition changes can enable/disable critical
| 518 |         md.push_str("## UDS Conformance Summary\n\n"); | protocol or I/O critical; changes may impact external i
| 519 |         md.push_str(&format!("- passed: {}\n", c.passed)); | support logic; impact depends on neighboring block
| 520 |         for s in &c.services { | iteration logic; may alter timing, throughput, and safety constraints. |
| 521 |             md.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 522 |                 "- SID {}: passed={} ({})\n", | support logic; impact depends on neighboring block and data flo
| 523 |                 s.service_sid, s.passed, s.evidence | support logic; impact depends on neighboring block and dat
| 524 |             )); | support logic; impact depends on neighboring block and data flow. |
| 525 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 526 |         md.push_str("\n"); | support logic; impact depends on neighboring block and data flow. |
| 527 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 528 |     md.push_str("## Phase Results\n\n"); | support logic; impact depends on neighboring block and data flow. |
| 529 |     for p in &report.phases { | iteration logic; may alter timing, throughput, and safety constraints. |
| 530 |         md.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 531 |             "- Fase {} - {}: {:?} ({})\n", | error propagation point; changes alter failure propagation semantic
| 532 |             p.phase, p.title, p.status, p.details | support logic; impact depends on neighboring block and data
| 533 |         )); | support logic; impact depends on neighboring block and data flow. |
| 534 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 535 |     fs::write(&md_path, md) | protocol or I/O critical; changes may impact external integration correctness. |
| 536 |         .map_err(\|e\| HwError::Unknown(format!("write report markdown failed: {e}")))?; | error propagation poi
| 537 | (blank) | blank separator; no runtime behavior. |
| 538 |     let json_bytes = fs::read(&json_path) | support logic; impact depends on neighboring block and data flow. |
| 539 |         .map_err(\|e\| HwError::Unknown(format!("read report json failed: {e}")))?; | error propagation point;
| 540 |     let mut h = Sha256::new(); | support logic; impact depends on neighboring block and data flow. |
| 541 |     h.update(&json_bytes); | support logic; impact depends on neighboring block and data flow. |
| 542 |     let report_hash = hex_sha(h.finalize().as_slice()); | support logic; impact depends on neighboring block and
| 543 | (blank) | blank separator; no runtime behavior. |
| 544 |     let mut report_final = report; | support logic; impact depends on neighboring block and data flow. |
| 545 |     if let Some(p11) = report_final.phases.iter_mut().find(\|p\| p.phase == 11) { | runtime gate; condition cha
| 546 |         p11.status = if Path::new(&json_path).exists() && Path::new(&md_path).exists() { | support logic; impac
| 547 |             PhaseStatus::Passed | support logic; impact depends on neighboring block and data flow. |
| 548 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 549 |             PhaseStatus::Failed | support logic; impact depends on neighboring block and data flow. |
| 550 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 551 |         p11.details = format!( | support logic; impact depends on neighboring block and data flow. |
| 552 |             "json={} md={} sha256={}", | support logic; impact depends on neighboring block and data flow. |
| 553 |             json_path.display(), | support logic; impact depends on neighboring block and data flow. |
| 554 |             md_path.display(), | support logic; impact depends on neighboring block and data flow. |
| 555 |             report_hash | support logic; impact depends on neighboring block and data flow. |
| 556 |         ); | support logic; impact depends on neighboring block and data flow. |
```

```
| 557 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 558 |     report_final.report_sha256 = Some(report_hash); | support logic; impact depends on neighboring block and dat
| 559 | (blank) | blank separator; no runtime behavior. |
| 560 |     let json_final = serde_json::to_string_pretty(&report_final) | support logic; impact depends on neighboring
| 561 |         .map_err(\|e\| HwError::Unknown(format!("serialize final report json failed: {e}")))?; | error propagati
| 562 |     fs::write(&json_path, json_final) | protocol or I/O critical; changes may impact external integration corre
| 563 |         .map_err(\|e\| HwError::Unknown(format!("write final report json failed: {e}")))?; | error propagation p
| 564 | (blank) | blank separator; no runtime behavior. |
| 565 |     Ok(json_path) | support logic; impact depends on neighboring block and data flow. |
| 566 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 567 | (blank) | blank separator; no runtime behavior. |
| 568 | fn run_sim_baseline_check() -> (bool, String) { | function signature/body; changes affect API behavior and contr
| 569 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 570 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 571 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 572 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 573 |     sim.throttle_pct = 35.0; | support logic; impact depends on neighboring block and data flow. |
| 574 |     for _ in 0..360 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 575 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 576 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 577 |     let ok = sim.elapsed > 5.0 && sim.metrics.steps_completed > 0; | support logic; impact depends on neighborin
| 578 |     ( | support logic; impact depends on neighboring block and data flow. |
| 579 |         ok, | support logic; impact depends on neighboring block and data flow. |
| 580 |         format!( | support logic; impact depends on neighboring block and data flow. |
| 581 |             "elapsed={:.2}s steps={} speed={:.2}kmh", | support logic; impact depends on neighboring block and d
| 582 |             sim.elapsed, sim.metrics.steps_completed, sim.tcm.ground_speed_kmh | support logic; impact depends o
| 583 |         ), | support logic; impact depends on neighboring block and data flow. |
| 584 |     ) | support logic; impact depends on neighboring block and data flow. |
| 585 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 586 | (blank) | blank separator; no runtime behavior. |
| 587 | fn run_sim_connectivity_check() -> (bool, String) { | function signature/body; changes affect API behavior and c
| 588 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 589 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 590 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 591 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 592 |     sim.throttle_pct = 20.0; | support logic; impact depends on neighboring block and data flow. |
| 593 |     for _ in 0..240 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 594 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 595 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 596 |     let can_health = sim.can_net.network_health_score_01(); | support logic; impact depends on neighboring bloc
| 597 |     let ok = (0.0..=1.0).contains(&can_health); | support logic; impact depends on neighboring block and data f
| 598 |     (ok, format!("can_health={:.3} errors={}", can_health, sim.can_net.total_errors())) | support logic; impact
| 599 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 600 | (blank) | blank separator; no runtime behavior. |
| 601 | fn run_sim_identity_check() -> (bool, String) { | function signature/body; changes affect API behavior and contr
| 602 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 603 |     let vin = sim.uds_ecm.process(&[0x22, 0xF1, 0x90], 0.0); | protocol or I/O critical; changes may impact exte
| 604 |     let sw = sim.uds_ecm.process(&[0x22, 0xF1, 0x89], 0.1); | protocol or I/O critical; changes may impact exter
| 605 |     let vin_ok = vin.first().copied() == Some(0x62) && vin.len() > 6; | support logic; impact depends on neighbo
| 606 |     let sw_ok = sw.first().copied() == Some(0x62) && sw.len() > 6; | support logic; impact depends on neighborin
| 607 |     ( | support logic; impact depends on neighboring block and data flow. |
| 608 |         vin_ok && sw_ok, | support logic; impact depends on neighboring block and data flow. |
| 609 |         format!("vin_ok={} sw_ok={} vin_len={} sw_len={}", vin_ok, sw_ok, vin.len(), sw.len()), | support logic
| 610 |     ) | support logic; impact depends on neighboring block and data flow. |
| 611 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 612 | (blank) | blank separator; no runtime behavior. |
| 613 | fn build_conformance_summary( | function signature/body; changes affect API behavior and control-flow. |
| 614 |     execute_flash: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 615 |     strict_mode: bool, | support logic; impact depends on neighboring block and data flow. |
| 616 |     flash_summary: &Option<FlashSummary>, | protocol or I/O critical; changes may impact external integration co
| 617 |     capabilities: &HwCapabilities, | support logic; impact depends on neighboring block and data flow. |
| 618 | ) -> ConformanceSummary { | support logic; impact depends on neighboring block and data flow. |
| 619 |     if !execute_flash { | runtime gate; condition changes can enable/disable critical paths. |
| 620 |         return ConformanceSummary { | support logic; impact depends on neighboring block and data flow. |
| 621 |             passed: !strict_mode, | support logic; impact depends on neighboring block and data flow. |
| 622 |             services: vec![ | support logic; impact depends on neighboring block and data flow. |
| 623 |                 ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow.
| 624 |                     service_sid: "0x27".to_string(), | support logic; impact depends on neighboring block and da
| 625 |                     passed: !strict_mode, | support logic; impact depends on neighboring block and data flow. |
| 626 |                     evidence: "flash path not executed".to_string(), | protocol or I/O critical; changes may imp
| 627 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 628 |                 ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow.
| 629 |                     service_sid: "0x34".to_string(), | support logic; impact depends on neighboring block and da
| 630 |                     passed: !strict_mode, | support logic; impact depends on neighboring block and data flow. |
| 631 |                     evidence: "flash path not executed".to_string(), | protocol or I/O critical; changes may imp
| 632 |                 }, | support logic; impact depends on neighboring block and data flow. |
```

```
| 633 |                 ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow.
| 634 |                     service_sid: "0x36".to_string(), | support logic; impact depends on neighboring block and da
| 635 |                     passed: !strict_mode, | support logic; impact depends on neighboring block and data flow. |
| 636 |                     evidence: "flash path not executed".to_string(), | protocol or I/O critical; changes may imp
| 637 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 638 |                 ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow.
| 639 |                     service_sid: "0x37".to_string(), | support logic; impact depends on neighboring block and da
| 640 |                     passed: !strict_mode, | support logic; impact depends on neighboring block and data flow. |
| 641 |                     evidence: "flash path not executed".to_string(), | protocol or I/O critical; changes may imp
| 642 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 643 |             ], | support logic; impact depends on neighboring block and data flow. |
| 644 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 645 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 646 | (blank) | blank separator; no runtime behavior. |
| 647 |     if let Some(sum) = flash_summary { | runtime gate; condition changes can enable/disable critical paths. |
| 648 |         let d = &sum.transport_diagnostics; | support logic; impact depends on neighboring block and data flow.
| 649 |         let sid27_ok = d.security_seed_positive && d.security_unlock_positive; | support logic; impact depends o
| 650 |         let sid34_ok = d.request_download_positive; | support logic; impact depends on neighboring block and dat
| 651 |         let sid36_ok = d.transfer_data_blocks_attempted > 0 | support logic; impact depends on neighboring block
| 652 |             && d.transfer_data_blocks_attempted == d.transfer_data_blocks_acked | support logic; impact depends
| 653 |             && d.sequence_error_count == 0 | support logic; impact depends on neighboring block and data flow.
| 654 |             && d.flowcontrol_timeout_count == 0; | support logic; impact depends on neighboring block and data f
| 655 |         let sid37_ok = d.request_transfer_exit_positive; | support logic; impact depends on neighboring block an
| 656 | (blank) | blank separator; no runtime behavior. |
| 657 |         let services = vec![ | support logic; impact depends on neighboring block and data flow. |
| 658 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 659 |                 service_sid: "0x27".to_string(), | support logic; impact depends on neighboring block and data f
| 660 |                 passed: sid27_ok, | support logic; impact depends on neighboring block and data flow. |
| 661 |                 evidence: format!( | support logic; impact depends on neighboring block and data flow. |
| 662 |                     "seed_ok={} unlock_ok={}", | support logic; impact depends on neighboring block and data flo
| 663 |                     d.security_seed_positive, d.security_unlock_positive | support logic; impact depends on neig
| 664 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 665 |             }, | support logic; impact depends on neighboring block and data flow. |
| 666 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 667 |                 service_sid: "0x34".to_string(), | support logic; impact depends on neighboring block and data f
| 668 |                 passed: sid34_ok, | support logic; impact depends on neighboring block and data flow. |
| 669 |                 evidence: format!("request_download_positive={}", d.request_download_positive), | support logic
| 670 |             }, | support logic; impact depends on neighboring block and data flow. |
| 671 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 672 |                 service_sid: "0x36".to_string(), | support logic; impact depends on neighboring block and data f
| 673 |                 passed: sid36_ok, | support logic; impact depends on neighboring block and data flow. |
| 674 |                 evidence: format!( | support logic; impact depends on neighboring block and data flow. |
| 675 |                     "acked={}/{} seq_err={} fc_timeout={} stmin_ms={:?} bs={:?}", | error propagation point; cha
| 676 |                     d.transfer_data_blocks_acked, | support logic; impact depends on neighboring block and data
| 677 |                     d.transfer_data_blocks_attempted, | support logic; impact depends on neighboring block and d
| 678 |                     d.sequence_error_count, | support logic; impact depends on neighboring block and data flow.
| 679 |                     d.flowcontrol_timeout_count, | support logic; impact depends on neighboring block and data f
| 680 |                     d.fc_stmin_seen_ms, | support logic; impact depends on neighboring block and data flow. |
| 681 |                     d.fc_blocksize_seen | support logic; impact depends on neighboring block and data flow. |
| 682 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 683 |             }, | support logic; impact depends on neighboring block and data flow. |
| 684 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 685 |                 service_sid: "0x37".to_string(), | support logic; impact depends on neighboring block and data f
| 686 |                 passed: sid37_ok, | support logic; impact depends on neighboring block and data flow. |
| 687 |                 evidence: format!("request_transfer_exit_positive={}", d.request_transfer_exit_positive), | supp
| 688 |             }, | support logic; impact depends on neighboring block and data flow. |
| 689 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 690 | (blank) | blank separator; no runtime behavior. |
| 691 |         return ConformanceSummary { | support logic; impact depends on neighboring block and data flow. |
| 692 |             passed: services.iter().all(\|s\| s.passed), | support logic; impact depends on neighboring block an
| 693 |             services, | support logic; impact depends on neighboring block and data flow. |
| 694 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 695 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 696 | (blank) | blank separator; no runtime behavior. |
| 697 |     ConformanceSummary { | support logic; impact depends on neighboring block and data flow. |
| 698 |         passed: false, | support logic; impact depends on neighboring block and data flow. |
| 699 |         services: vec![ | support logic; impact depends on neighboring block and data flow. |
| 700 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 701 |                 service_sid: "0x27".to_string(), | support logic; impact depends on neighboring block and data f
| 702 |                 passed: false, | support logic; impact depends on neighboring block and data flow. |
| 703 |                 evidence: format!( | support logic; impact depends on neighboring block and data flow. |
| 704 |                     "flash path requested without summary; capabilities={}", | protocol or I/O critical; changes
| 705 |                     if capabilities.missing_for_live_flash().is_empty() { | runtime gate; condition changes can
| 706 |                         "present".to_string() | support logic; impact depends on neighboring block and data flo
| 707 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 708 |                         capabilities.missing_for_live_flash().join(",") | protocol or I/O critical; changes may
```

```
| 709 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 710 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 711 |             }, | support logic; impact depends on neighboring block and data flow. |
| 712 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 713 |                 service_sid: "0x34".to_string(), | support logic; impact depends on neighboring block and data f
| 714 |                 passed: false, | support logic; impact depends on neighboring block and data flow. |
| 715 |                 evidence: "flash summary unavailable".to_string(), | protocol or I/O critical; changes may impac
| 716 |             }, | support logic; impact depends on neighboring block and data flow. |
| 717 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 718 |                 service_sid: "0x36".to_string(), | support logic; impact depends on neighboring block and data f
| 719 |                 passed: false, | support logic; impact depends on neighboring block and data flow. |
| 720 |                 evidence: "flash summary unavailable".to_string(), | protocol or I/O critical; changes may impac
| 721 |             }, | support logic; impact depends on neighboring block and data flow. |
| 722 |             ServiceConformanceResult { | support logic; impact depends on neighboring block and data flow. |
| 723 |                 service_sid: "0x37".to_string(), | support logic; impact depends on neighboring block and data f
| 724 |                 passed: false, | support logic; impact depends on neighboring block and data flow. |
| 725 |                 evidence: "flash summary unavailable".to_string(), | protocol or I/O critical; changes may impac
| 726 |             }, | support logic; impact depends on neighboring block and data flow. |
| 727 |         ], | support logic; impact depends on neighboring block and data flow. |
| 728 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 729 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 730 | (blank) | blank separator; no runtime behavior. |
| 731 | fn run_sim_preflight_check() -> (bool, String) { | function signature/body; changes affect API behavior and cont
| 732 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 733 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 734 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 735 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 736 |     for _ in 0..120 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 737 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 738 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 739 |     let v = sim.bcm.battery_voltage; | support logic; impact depends on neighboring block and data flow. |
| 740 |     let ok = v >= 11.8; | support logic; impact depends on neighboring block and data flow. |
| 741 |     (ok, format!("battery_v={:.2}", v)) | support logic; impact depends on neighboring block and data flow. |
| 742 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 743 | (blank) | blank separator; no runtime behavior. |
| 744 | fn run_uds_runtime_self_test() -> (bool, String) { | function signature/body; changes affect API behavior and co
| 745 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 746 |     let sess = sim.uds_ecm.process(&[0x10, 0x01], 0.0); | protocol or I/O critical; changes may impact external
| 747 |     let vin = sim.uds_ecm.process(&[0x22, 0xF1, 0x90], 0.1); | protocol or I/O critical; changes may impact exte
| 748 |     let tester = sim.uds_ecm.process(&[0x3E, 0x00], 0.2); | protocol or I/O critical; changes may impact externa
| 749 |     let ok = sess.first().copied() == Some(0x50) | support logic; impact depends on neighboring block and data f
| 750 |         && vin.first().copied() == Some(0x62) | support logic; impact depends on neighboring block and data flo
| 751 |         && tester.first().copied() == Some(0x7E); | support logic; impact depends on neighboring block and data
| 752 |     ( | support logic; impact depends on neighboring block and data flow. |
| 753 |         ok, | support logic; impact depends on neighboring block and data flow. |
| 754 |         format!( | support logic; impact depends on neighboring block and data flow. |
| 755 |             "session_sid=0x{:02X} vin_sid=0x{:02X} tester_sid=0x{:02X}", | support logic; impact depends on neig
| 756 |             sess.first().copied().unwrap_or(0), | support logic; impact depends on neighboring block and data f
| 757 |             vin.first().copied().unwrap_or(0), | support logic; impact depends on neighboring block and data flo
| 758 |             tester.first().copied().unwrap_or(0) | support logic; impact depends on neighboring block and data f
| 759 |         ), | support logic; impact depends on neighboring block and data flow. |
| 760 |     ) | support logic; impact depends on neighboring block and data flow. |
| 761 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 762 | (blank) | blank separator; no runtime behavior. |
| 763 | fn run_sim_stress_check() -> (bool, String) { | function signature/body; changes affect API behavior and contro
| 764 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 765 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 766 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 767 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 768 |     let mut max_speed = 0.0_f64; | support logic; impact depends on neighboring block and data flow. |
| 769 |     for i in 0..1800 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 770 |         sim.throttle_pct = if i % 300 < 180 { 55.0 } else { 15.0 }; | support logic; impact depends on neighbori
| 771 |         sim.brake_pct = if i % 450 > 400 { 20.0 } else { 0.0 }; | support logic; impact depends on neighboring
| 772 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 773 |         max_speed = max_speed.max(sim.tcm.ground_speed_kmh); | support logic; impact depends on neighboring bloc
| 774 |         if sim.tcm.ground_speed_kmh.is_nan() { | runtime gate; condition changes can enable/disable critical pa
| 775 |             return (false, "speed NaN detected".to_string()); | support logic; impact depends on neighboring bl
| 776 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 777 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 778 |     ( | support logic; impact depends on neighboring block and data flow. |
| 779 |         true, | support logic; impact depends on neighboring block and data flow. |
| 780 |         format!("max_speed={:.2} fuel={:.2}%", max_speed, sim.ecm.fuel_level_pct), | support logic; impact depe
| 781 |     ) | support logic; impact depends on neighboring block and data flow. |
| 782 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 783 | (blank) | blank separator; no runtime behavior. |
| 784 | fn hex_sha(bytes: &[u8]) -> String { | function signature/body; changes affect API behavior and control-flow. |
```

```
| 785 |     let mut out = String::with_capacity(bytes.len() * 2); | support logic; impact depends on neighboring block a
| 786 |     for b in bytes { | iteration logic; may alter timing, throughput, and safety constraints. |
| 787 |         out.push_str(&format!("{b:02x}")); | support logic; impact depends on neighboring block and data flow.
| 788 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 789 |     out | support logic; impact depends on neighboring block and data flow. |
| 790 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 791 | (blank) | blank separator; no runtime behavior. |
| 792 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 793 |     SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 794 |         .duration_since(UNIX_EPOCH) | support logic; impact depends on neighboring block and data flow. |
| 795 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 796 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 797 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/io/rate_limiter.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 85 | Non-empty: 73 | Symbol count: 12

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 6 |
| try_take | 4 |
| check_can | 3 |
| refill | 2 |
| check_serial | 1 |
| limiter_blocks_when_empty | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 5 | struct | TokenBucket | data/schema coupling and match/serde break risk |
| 12 | impl | TokenBucket | method behavior and trait semantics |
| 13 | fn | new | API and behavior coupling to callers/implementers |
| 23 | fn | try_take | API and behavior coupling to callers/implementers |
| 33 | fn | refill | API and behavior coupling to callers/implementers |
| 42 | struct | TwoLevelRateLimiter | data/schema coupling and match/serde break risk |
| 48 | impl | TwoLevelRateLimiter | method behavior and trait semantics |
| 49 | fn | new | API and behavior coupling to callers/implementers |
| 57 | fn | check_can | API and behavior coupling to callers/implementers |
| 70 | fn | check_serial | API and behavior coupling to callers/implementers |
| 76 | module | tests | namespace and compile-time linkage |
| 80 | fn | limiter_blocks_when_empty | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::collections::HashMap; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::time::Instant; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 5 | struct TokenBucket { | data layout contract; field changes can break callers/serde/tests. |
| 6 |     tokens: f64, | support logic; impact depends on neighboring block and data flow. |
| 7 |     capacity: f64, | support logic; impact depends on neighboring block and data flow. |
| 8 |     refill_per_sec: f64, | support logic; impact depends on neighboring block and data flow. |
| 9 |     last_refill: Instant, | support logic; impact depends on neighboring block and data flow. |
| 10 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | impl TokenBucket { | implementation binding; behavior semantics and trait conformance live here. |
| 13 |     fn new(rate_per_sec: u32, burst: u32) -> Self { | function signature/body; changes affect API behavior and co
| 14 |         let cap = burst.max(1) as f64; | support logic; impact depends on neighboring block and data flow. |
| 15 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 16 |             tokens: cap, | support logic; impact depends on neighboring block and data flow. |
| 17 |             capacity: cap, | support logic; impact depends on neighboring block and data flow. |
| 18 |             refill_per_sec: rate_per_sec.max(1) as f64, | support logic; impact depends on neighboring block and
| 19 |             last_refill: Instant::now(), | support logic; impact depends on neighboring block and data flow. |
| 20 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 21 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 22 | (blank) | blank separator; no runtime behavior. |
| 23 |     fn try_take(&mut self, amount: f64) -> bool { | function signature/body; changes affect API behavior and con
| 24 |         self.refill(); | support logic; impact depends on neighboring block and data flow. |
| 25 |         if self.tokens >= amount { | runtime gate; condition changes can enable/disable critical paths. |
| 26 |             self.tokens -= amount; | support logic; impact depends on neighboring block and data flow. |
| 27 |             true | support logic; impact depends on neighboring block and data flow. |
| 28 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 29 |             false | support logic; impact depends on neighboring block and data flow. |
| 30 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 31 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 32 | (blank) | blank separator; no runtime behavior. |
| 33 |     fn refill(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 34 |         let now = Instant::now(); | support logic; impact depends on neighboring block and data flow. |
| 35 |         let elapsed = now.duration_since(self.last_refill).as_secs_f64(); | support logic; impact depends on neig
| 36 |         self.last_refill = now; | support logic; impact depends on neighboring block and data flow. |
| 37 |         self.tokens = (self.tokens + elapsed * self.refill_per_sec).min(self.capacity); | support logic; impact c
| 38 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 40 | (blank) | blank separator; no runtime behavior. |
| 41 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 42 | pub struct TwoLevelRateLimiter { | data layout contract; field changes can break callers/serde/tests. |
| 43 |     global: TokenBucket, | support logic; impact depends on neighboring block and data flow. |
| 44 |     per_id_rate: u32, | support logic; impact depends on neighboring block and data flow. |
| 45 |     per_id: HashMap<u32, TokenBucket>, | support logic; impact depends on neighboring block and data flow. |
| 46 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 | (blank) | blank separator; no runtime behavior. |
| 48 | impl TwoLevelRateLimiter { | implementation binding; behavior semantics and trait conformance live here. |
| 49 |     pub fn new(global_per_sec: u32, per_id_per_sec: u32) -> Self { | function signature/body; changes affect API
| 50 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 51 |             global: TokenBucket::new(global_per_sec, global_per_sec), | support logic; impact depends on neighbo
| 52 |             per_id_rate: per_id_per_sec.max(1), | support logic; impact depends on neighboring block and data flo
| 53 |             per_id: HashMap::new(), | support logic; impact depends on neighboring block and data flow. |
| 54 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 56 | (blank) | blank separator; no runtime behavior. |
| 57 |     pub fn check_can(&mut self, can_id: u32) -> bool { | function signature/body; changes affect API behavior and
| 58 |         if !self.global.try_take(1.0) { | runtime gate; condition changes can enable/disable critical paths. |
| 59 |             return false; | support logic; impact depends on neighboring block and data flow. |
| 60 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 61 | (blank) | blank separator; no runtime behavior. |
| 62 |         let per = self | support logic; impact depends on neighboring block and data flow. |
| 63 |             .per_id | support logic; impact depends on neighboring block and data flow. |
| 64 |             .entry(can_id) | support logic; impact depends on neighboring block and data flow. |
| 65 |             .or_insert_with(\|\| TokenBucket::new(self.per_id_rate, self.per_id_rate)); | support logic; impact c
| 66 | (blank) | blank separator; no runtime behavior. |
| 67 |         per.try_take(1.0) | support logic; impact depends on neighboring block and data flow. |
| 68 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 69 | (blank) | blank separator; no runtime behavior. |
| 70 |     pub fn check_serial(&mut self) -> bool { | function signature/body; changes affect API behavior and control-
| 71 |         self.global.try_take(1.0) | support logic; impact depends on neighboring block and data flow. |
| 72 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 73 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 74 | (blank) | blank separator; no runtime behavior. |
| 75 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 76 | mod tests { | module declaration; changing can break namespace graph. |
| 77 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 78 | (blank) | blank separator; no runtime behavior. |
| 79 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 80 |     fn limiter_blocks_when_empty() { | function signature/body; changes affect API behavior and control-flow. |
| 81 |         let mut lim = TwoLevelRateLimiter::new(1, 1); | support logic; impact depends on neighboring block and da
| 82 |         assert!(lim.check_can(0x100)); | support logic; impact depends on neighboring block and data flow. |
| 83 |         assert!(!lim.check_can(0x100)); | support logic; impact depends on neighboring block and data flow. |
| 84 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 85 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/io/replay.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 183 | Non-empty: 165 | Symbol count: 11

### Function Call Hotspots

```
| Function | Approx Calls In File |
```

```
|---|---:|
| len | 5 |
| new | 4 |
| push | 4 |
| hex_encode | 3 |
| hex_decode | 3 |
| nibble | 3 |
| open | 2 |
| frame_to_record | 2 |
| record_to_frame | 2 |
| append_record | 1 |
| read_records | 1 |
| now_ms | 1 |
| can_record_roundtrip | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 10 | struct | JsonlRecord | data/schema coupling and match/serde break risk |
| 22 | fn | append_record | API and behavior coupling to callers/implementers |
| 39 | fn | read_records | API and behavior coupling to callers/implementers |
| 55 | fn | frame_to_record | API and behavior coupling to callers/implementers |
| 87 | fn | record_to_frame | API and behavior coupling to callers/implementers |
| 121 | fn | now_ms | API and behavior coupling to callers/implementers |
| 128 | fn | hex_encode | API and behavior coupling to callers/implementers |
| 137 | fn | hex_decode | API and behavior coupling to callers/implementers |
| 151 | fn | nibble | API and behavior coupling to callers/implementers |
| 159 | module | tests | namespace and compile-time linkage |
| 163 | fn | can_record_roundtrip | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::fs::{self, OpenOptions}; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::io::{BufRead, BufReader, Write}; | import wiring; changing path/name can break compilation and module lin
| 3 | use std::path::Path; | import wiring; changing path/name can break compilation and module linkage. |
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | use super::hw::{CanFrame, Frame, HwError, SerialFrame}; | import wiring; changing path/name can break compilation
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | #[derive(Debug, Clone, Serialize, Deserialize, PartialEq, Eq)] | comment/documentation; changing affects understan
| 10 | pub struct JsonlRecord { | data layout contract; field changes can break callers/serde/tests. |
| 11 |     pub ts: u64, | support logic; impact depends on neighboring block and data flow. |
| 12 |     pub transport: String, | support logic; impact depends on neighboring block and data flow. |
| 13 |     pub dir: String, | support logic; impact depends on neighboring block and data flow. |
| 14 |     pub id: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 15 |     pub dlc: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub data: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 17 |     pub raw_hex: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 18 |     pub allowed: Option<bool>, | support logic; impact depends on neighboring block and data flow. |
| 19 |     pub dry_run: Option<bool>, | support logic; impact depends on neighboring block and data flow. |
| 20 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 21 | (blank) | blank separator; no runtime behavior. |
| 22 | pub fn append_record(path: &Path, rec: &JsonlRecord) -> Result<(), HwError> { | function signature/body; changes
| 23 |     if let Some(parent) = path.parent() { | runtime gate; condition changes can enable/disable critical paths. |
| 24 |         fs::create_dir_all(parent) | support logic; impact depends on neighboring block and data flow. |
| 25 |             .map_err(\|e\| HwError::Unknown(format!("create log dir failed: {e}")))?; | error propagation point;
| 26 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 | (blank) | blank separator; no runtime behavior. |
| 28 |     let mut f = OpenOptions::new() | support logic; impact depends on neighboring block and data flow. |
| 29 |         .create(true) | support logic; impact depends on neighboring block and data flow. |
| 30 |         .append(true) | support logic; impact depends on neighboring block and data flow. |
| 31 |         .open(path) | support logic; impact depends on neighboring block and data flow. |
| 32 |         .map_err(\|e\| HwError::Unknown(format!("open jsonl log failed: {e}")))?; | error propagation point; char
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 |     let line = serde_json::to_string(rec) | support logic; impact depends on neighboring block and data flow. |
| 35 |         .map_err(\|e\| HwError::Unknown(format!("json encode failed: {e}")))?; | error propagation point; changes
| 36 |     writeln!(f, "{line}").map_err(\|e\| HwError::Unknown(format!("write jsonl failed: {e}"))) | protocol or I/O
| 37 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | (blank) | blank separator; no runtime behavior. |
| 39 | pub fn read_records(path: &Path) -> Result<Vec<JsonlRecord>, HwError> { | function signature/body; changes affec
| 40 |     let f = OpenOptions::new() | support logic; impact depends on neighboring block and data flow. |
```

```
| 41 |         .read(true) | support logic; impact depends on neighboring block and data flow. |
| 42 |         .open(path) | support logic; impact depends on neighboring block and data flow. |
| 43 |         .map_err(\|e\| HwError::Unknown(format!("open replay file failed: {e}")))?; | error propagation point; ch
| 44 |     let r = BufReader::new(f); | support logic; impact depends on neighboring block and data flow. |
| 45 |     let mut out = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 46 |     for line in r.lines() { | iteration logic; may alter timing, throughput, and safety constraints. |
| 47 |         let line = line.map_err(\|e\| HwError::Unknown(format!("read line failed: {e}")))?; | error propagation p
| 48 |         let rec = serde_json::from_str::<JsonlRecord>(&line) | support logic; impact depends on neighboring block
| 49 |             .map_err(\|e\| HwError::Unknown(format!("parse line failed: {e}")))?; | error propagation point; char
| 50 |         out.push(rec); | support logic; impact depends on neighboring block and data flow. |
| 51 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 |     Ok(out) | support logic; impact depends on neighboring block and data flow. |
| 53 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 | (blank) | blank separator; no runtime behavior. |
| 55 | pub fn frame_to_record( | function signature/body; changes affect API behavior and control-flow. |
| 56 |     frame: &Frame, | support logic; impact depends on neighboring block and data flow. |
| 57 |     dir: &str, | support logic; impact depends on neighboring block and data flow. |
| 58 |     allowed: Option<bool>, | support logic; impact depends on neighboring block and data flow. |
| 59 |     dry_run: Option<bool>, | support logic; impact depends on neighboring block and data flow. |
| 60 | ) -> JsonlRecord { | support logic; impact depends on neighboring block and data flow. |
| 61 |     match frame { | branch dispatcher; arm changes alter behavior class handling. |
| 62 |         Frame::Can(cf) => JsonlRecord { | support logic; impact depends on neighboring block and data flow. |
| 63 |             ts: cf.timestamp_ms.unwrap_or_else(now_ms), | support logic; impact depends on neighboring block and
| 64 |             transport: "can".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 65 |             dir: dir.to_string(), | support logic; impact depends on neighboring block and data flow. |
| 66 |             id: Some(cf.id), | support logic; impact depends on neighboring block and data flow. |
| 67 |             dlc: Some(cf.dlc), | support logic; impact depends on neighboring block and data flow. |
| 68 |             data: Some(hex_encode(&cf.data[..cf.len.min(8)])), | support logic; impact depends on neighboring bl
| 69 |             raw_hex: None, | support logic; impact depends on neighboring block and data flow. |
| 70 |             allowed, | support logic; impact depends on neighboring block and data flow. |
| 71 |             dry_run, | support logic; impact depends on neighboring block and data flow. |
| 72 |         }, | support logic; impact depends on neighboring block and data flow. |
| 73 |         Frame::Serial(sf) => JsonlRecord { | support logic; impact depends on neighboring block and data flow. |
| 74 |             ts: sf.timestamp_ms.unwrap_or_else(now_ms), | support logic; impact depends on neighboring block and
| 75 |             transport: "serial".to_string(), | support logic; impact depends on neighboring block and data flow.
| 76 |             dir: dir.to_string(), | support logic; impact depends on neighboring block and data flow. |
| 77 |             id: None, | support logic; impact depends on neighboring block and data flow. |
| 78 |             dlc: Some((sf.bytes.len().min(255)) as u8), | support logic; impact depends on neighboring block and
| 79 |             data: Some(hex_encode(&sf.bytes)), | support logic; impact depends on neighboring block and data flo
| 80 |             raw_hex: None, | support logic; impact depends on neighboring block and data flow. |
| 81 |             allowed, | support logic; impact depends on neighboring block and data flow. |
| 82 |             dry_run, | support logic; impact depends on neighboring block and data flow. |
| 83 |         }, | support logic; impact depends on neighboring block and data flow. |
| 84 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 85 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 86 | (blank) | blank separator; no runtime behavior. |
| 87 | pub fn record_to_frame(rec: &JsonlRecord) -> Option<Frame> { | function signature/body; changes affect API behavi
| 88 |     if rec.dir != "rx" { | runtime gate; condition changes can enable/disable critical paths. |
| 89 |         return None; | support logic; impact depends on neighboring block and data flow. |
| 90 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 91 | (blank) | blank separator; no runtime behavior. |
| 92 |     if rec.transport == "can" { | runtime gate; condition changes can enable/disable critical paths. |
| 93 |         let id = rec.id?; | error propagation point; changes alter failure propagation semantics. |
| 94 |         let data_hex = rec.data.as_deref()?; | error propagation point; changes alter failure propagation semanti
| 95 |         let bytes = hex_decode(data_hex)?; | error propagation point; changes alter failure propagation semantics
| 96 |         let mut data = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 97 |         let len = bytes.len().min(8); | support logic; impact depends on neighboring block and data flow. |
| 98 |         data[..len].copy_from_slice(&bytes[..len]); | support logic; impact depends on neighboring block and data
| 99 |         return Some(Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 100 |             id, | support logic; impact depends on neighboring block and data flow. |
| 101 |             dlc: len as u8, | support logic; impact depends on neighboring block and data flow. |
| 102 |             data, | support logic; impact depends on neighboring block and data flow. |
| 103 |             len, | support logic; impact depends on neighboring block and data flow. |
| 104 |             timestamp_ms: Some(rec.ts), | support logic; impact depends on neighboring block and data flow. |
| 105 |         })); | support logic; impact depends on neighboring block and data flow. |
| 106 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 107 | (blank) | blank separator; no runtime behavior. |
| 108 |     if rec.transport == "serial" { | runtime gate; condition changes can enable/disable critical paths. |
| 109 |         let data_hex = rec.data.as_deref()?; | error propagation point; changes alter failure propagation seman
| 110 |         let bytes = hex_decode(data_hex)?; | error propagation point; changes alter failure propagation semantic
| 111 |         return Some(Frame::Serial(SerialFrame { | support logic; impact depends on neighboring block and data f
| 112 |             bytes, | support logic; impact depends on neighboring block and data flow. |
| 113 |             protocol_hint: None, | support logic; impact depends on neighboring block and data flow. |
| 114 |             timestamp_ms: Some(rec.ts), | support logic; impact depends on neighboring block and data flow. |
| 115 |         })); | support logic; impact depends on neighboring block and data flow. |
| 116 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 117 | (blank) | blank separator; no runtime behavior. |
| 118 |     None | support logic; impact depends on neighboring block and data flow. |
| 119 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 120 | (blank) | blank separator; no runtime behavior. |
| 121 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 122 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 123 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data flo
| 124 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 125 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 126 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 127 | (blank) | blank separator; no runtime behavior. |
| 128 | fn hex_encode(bytes: &[u8]) -> String { | function signature/body; changes affect API behavior and control-flow
| 129 |     let mut out = String::with_capacity(bytes.len() * 2); | support logic; impact depends on neighboring block a
| 130 |     for b in bytes { | iteration logic; may alter timing, throughput, and safety constraints. |
| 131 |         out.push(nibble((*b >> 4) & 0x0F)); | support logic; impact depends on neighboring block and data flow.
| 132 |         out.push(nibble(*b & 0x0F)); | support logic; impact depends on neighboring block and data flow. |
| 133 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 134 |     out | support logic; impact depends on neighboring block and data flow. |
| 135 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 136 | (blank) | blank separator; no runtime behavior. |
| 137 | fn hex_decode(hex: &str) -> Option<Vec<u8>> { | function signature/body; changes affect API behavior and contro
| 138 |     if !hex.len().is_multiple_of(2) { | runtime gate; condition changes can enable/disable critical paths. |
| 139 |         return None; | support logic; impact depends on neighboring block and data flow. |
| 140 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 141 |     let mut out = Vec::with_capacity(hex.len() / 2); | support logic; impact depends on neighboring block and da
| 142 |     let mut chars = hex.chars(); | support logic; impact depends on neighboring block and data flow. |
| 143 |     while let (Some(h), Some(l)) = (chars.next(), chars.next()) { | iteration logic; may alter timing, throughpu
| 144 |         let hi = h.to_digit(16)? as u8; | error propagation point; changes alter failure propagation semantics.
| 145 |         let lo = l.to_digit(16)? as u8; | error propagation point; changes alter failure propagation semantics.
| 146 |         out.push((hi << 4) \| lo); | support logic; impact depends on neighboring block and data flow. |
| 147 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 148 |     Some(out) | support logic; impact depends on neighboring block and data flow. |
| 149 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 150 | (blank) | blank separator; no runtime behavior. |
| 151 | fn nibble(v: u8) -> char { | function signature/body; changes affect API behavior and control-flow. |
| 152 |     match v { | branch dispatcher; arm changes alter behavior class handling. |
| 153 |         0..=9 => (b'0' + v) as char, | support logic; impact depends on neighboring block and data flow. |
| 154 |         _ => (b'a' + (v - 10)) as char, | support logic; impact depends on neighboring block and data flow. |
| 155 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 156 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 157 | (blank) | blank separator; no runtime behavior. |
| 158 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 159 | mod tests { | module declaration; changing can break namespace graph. |
| 160 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 161 | (blank) | blank separator; no runtime behavior. |
| 162 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 163 |     fn can_record_roundtrip() { | function signature/body; changes affect API behavior and control-flow. |
| 164 |         let f = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 165 |             id: 0x123, | support logic; impact depends on neighboring block and data flow. |
| 166 |             dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 167 |             data: [1, 2, 3, 4, 5, 6, 7, 8], | support logic; impact depends on neighboring block and data flow.
| 168 |             len: 8, | support logic; impact depends on neighboring block and data flow. |
| 169 |             timestamp_ms: Some(12345), | support logic; impact depends on neighboring block and data flow. |
| 170 |         }); | support logic; impact depends on neighboring block and data flow. |
| 171 |         let rec = frame_to_record(&f, "rx", None, None); | support logic; impact depends on neighboring block an
| 172 |         let got = record_to_frame(&rec).expect("frame from record"); | panic boundary; edits alter crash behavio
| 173 |         assert_eq!(rec.ts, 12345); | support logic; impact depends on neighboring block and data flow. |
| 174 |         assert!(matches!(got, Frame::Can(_)), "expected can frame"); | support logic; impact depends on neighbor
| 175 |         if let Frame::Can(cf) = got { | runtime gate; condition changes can enable/disable critical paths. |
| 176 |             assert_eq!(cf.id, 0x123); | support logic; impact depends on neighboring block and data flow. |
| 177 |             assert_eq!(cf.dlc, 8); | support logic; impact depends on neighboring block and data flow. |
| 178 |             assert_eq!(cf.len, 8); | support logic; impact depends on neighboring block and data flow. |
| 179 |             assert_eq!(cf.data, [1, 2, 3, 4, 5, 6, 7, 8]); | support logic; impact depends on neighboring block
| 180 |             assert_eq!(cf.timestamp_ms, Some(12345)); | support logic; impact depends on neighboring block and
| 181 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 182 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 183 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/io/serial_adapter.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 96 | Non-empty: 81 | Symbol count: 10

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| open | 2 |
| read_frame | 2 |
| now_ms | 2 |
| new | 1 |
| init | 1 |
| try_read_frame | 1 |
| send_frame | 1 |
| flush | 1 |
| close | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 9 | struct | SerialAdapter | data/schema coupling and match/serde break risk |
| 13 | impl | SerialAdapter | method behavior and trait semantics |
| 14 | fn | open | API and behavior coupling to callers/implementers |
| 30 | impl | HardwareInterface for SerialAdapter | method behavior and trait semantics |
| 31 | fn | init | API and behavior coupling to callers/implementers |
| 35 | fn | read_frame | API and behavior coupling to callers/implementers |
| 61 | fn | try_read_frame | API and behavior coupling to callers/implementers |
| 69 | fn | send_frame | API and behavior coupling to callers/implementers |
| 85 | fn | close | API and behavior coupling to callers/implementers |
| 91 | fn | now_ms | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::io::{Read, Write}; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::time::Duration; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use serialport::SerialPort; | import wiring; changing path/name can break compilation and module linkage. |
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use super::hw::{Frame, HardwareInterface, HwConfig, HwError, SerialFrame}; | import wiring; changing path/name can
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | #[derive(Debug, Default)] | comment/documentation; changing affects understanding and intent traceability. |
| 9 | pub struct SerialAdapter { | data layout contract; field changes can break callers/serde/tests. |
| 10 |     port: Option<Box<dyn SerialPort>>, | support logic; impact depends on neighboring block and data flow. |
| 11 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | impl SerialAdapter { | implementation binding; behavior semantics and trait conformance live here. |
| 14 |     pub fn open(port_name: &str, baud: u32) -> Result<Self, HwError> { | function signature/body; changes affect
| 15 |         let port = serialport::new(port_name, baud) | support logic; impact depends on neighboring block and data
| 16 |             .timeout(Duration::from_millis(250)) | support logic; impact depends on neighboring block and data f
| 17 |             .open() | support logic; impact depends on neighboring block and data flow. |
| 18 |             .map_err(\|e\| { | support logic; impact depends on neighboring block and data flow. |
| 19 |                 let msg = e.to_string(); | support logic; impact depends on neighboring block and data flow. |
| 20 |                 if msg.to_ascii_lowercase().contains("permission") { | runtime gate; condition changes can enable
| 21 |                     HwError::PermissionDenied(msg) | support logic; impact depends on neighboring block and data
| 22 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 23 |                     HwError::PortNotFound(msg) | support logic; impact depends on neighboring block and data flo
| 24 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 25 |             })?; | error propagation point; changes alter failure propagation semantics. |
| 26 |         Ok(Self { port: Some(port) }) | support logic; impact depends on neighboring block and data flow. |
| 27 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 28 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 29 | (blank) | blank separator; no runtime behavior. |
| 30 | impl HardwareInterface for SerialAdapter { | implementation binding; behavior semantics and trait conformance li
| 31 |     fn init(&mut self, _config: &HwConfig) -> Result<(), HwError> { | function signature/body; changes affect AP
| 32 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 33 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 |     fn read_frame(&mut self) -> Result<Frame, HwError> { | function signature/body; changes affect API behavior
| 36 |         let Some(port) = self.port.as_mut() else { | support logic; impact depends on neighboring block and data
| 37 |             return Err(HwError::PortNotFound("serial port not initialized".into())); | support logic; impact dep
| 38 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 | (blank) | blank separator; no runtime behavior. |
| 40 |         let mut buf = vec![0u8; 1024]; | support logic; impact depends on neighboring block and data flow. |
| 41 |         let n = port.read(&mut buf).map_err(\|e\| { | support logic; impact depends on neighboring block and data
| 42 |             if e.kind() == std::io::ErrorKind::TimedOut { | runtime gate; condition changes can enable/disable c
| 43 |                 HwError::Timeout | support logic; impact depends on neighboring block and data flow. |
```

```
| 44 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 45 |                 HwError::TransceiverError | support logic; impact depends on neighboring block and data flow. |
| 46 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 48 | (blank) | blank separator; no runtime behavior. |
| 49 |         if n == 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 50 |             return Err(HwError::Timeout); | support logic; impact depends on neighboring block and data flow. |
| 51 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 | (blank) | blank separator; no runtime behavior. |
| 53 |         buf.truncate(n); | support logic; impact depends on neighboring block and data flow. |
| 54 |         Ok(Frame::Serial(SerialFrame { | support logic; impact depends on neighboring block and data flow. |
| 55 |             bytes: buf, | support logic; impact depends on neighboring block and data flow. |
| 56 |             protocol_hint: Some("raw-serial".into()), | support logic; impact depends on neighboring block and da
| 57 |             timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 58 |         })) | support logic; impact depends on neighboring block and data flow. |
| 59 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 |     fn try_read_frame(&mut self) -> Result<Option<Frame>, HwError> { | function signature/body; changes affect AP
| 62 |         match self.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 63 |             Ok(f) => Ok(Some(f)), | support logic; impact depends on neighboring block and data flow. |
| 64 |             Err(HwError::Timeout) => Ok(None), | support logic; impact depends on neighboring block and data flow
| 65 |             Err(e) => Err(e), | support logic; impact depends on neighboring block and data flow. |
| 66 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 67 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 68 | (blank) | blank separator; no runtime behavior. |
| 69 |     fn send_frame(&mut self, frame: Frame) -> Result<(), HwError> { | function signature/body; changes affect API
| 70 |         let Some(port) = self.port.as_mut() else { | support logic; impact depends on neighboring block and data
| 71 |             return Err(HwError::PortNotFound("serial port not initialized".into())); | support logic; impact depe
| 72 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 73 | (blank) | blank separator; no runtime behavior. |
| 74 |         let bytes: Vec<u8> = match frame { | support logic; impact depends on neighboring block and data flow. |
| 75 |             Frame::Serial(sf) => sf.bytes, | support logic; impact depends on neighboring block and data flow. |
| 76 |             Frame::Can(cf) => cf.data[..cf.len.min(8)].to_vec(), | support logic; impact depends on neighboring b
| 77 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 78 | (blank) | blank separator; no runtime behavior. |
| 79 |         port.write_all(&bytes) | protocol or I/O critical; changes may impact external integration correctness.
| 80 |             .map_err(\|_e\| HwError::TransceiverError)?; | error propagation point; changes alter failure propaga
| 81 |         port.flush().map_err(\|_e\| HwError::TransceiverError)?; | error propagation point; changes alter failure
| 82 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 83 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 84 | (blank) | blank separator; no runtime behavior. |
| 85 |     fn close(&mut self) -> Result<(), HwError> { | function signature/body; changes affect API behavior and contr
| 86 |         self.port = None; | support logic; impact depends on neighboring block and data flow. |
| 87 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 88 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 90 | (blank) | blank separator; no runtime behavior. |
| 91 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 92 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 93 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data flow
| 94 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 95 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 96 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/io/socketcan_adapter.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 145 | Non-empty: 124 | Symbol count: 13

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| read_frame | 4 |
| open | 3 |
| now_ms | 2 |
| send_frame | 2 |
| new | 2 |
| init | 1 |
| len | 1 |
| try_read_frame | 1 |
| close | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 10 | struct | SocketCanAdapter | data/schema coupling and match/serde break risk |
| 15 | impl | SocketCanAdapter | method behavior and trait semantics |
| 17 | fn | open | API and behavior coupling to callers/implementers |
| 28 | fn | open | API and behavior coupling to callers/implementers |
| 35 | impl | HardwareInterface for SocketCanAdapter | method behavior and trait semantics |
| 36 | fn | init | API and behavior coupling to callers/implementers |
| 41 | fn | read_frame | API and behavior coupling to callers/implementers |
| 74 | fn | read_frame | API and behavior coupling to callers/implementers |
| 80 | fn | try_read_frame | API and behavior coupling to callers/implementers |
| 89 | fn | send_frame | API and behavior coupling to callers/implementers |
| 124 | fn | send_frame | API and behavior coupling to callers/implementers |
| 130 | fn | close | API and behavior coupling to callers/implementers |
| 140 | fn | now_ms | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use super::hw::{Frame, HardwareInterface, HwConfig, HwError}; | import wiring; changing path/name can break compil
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceability. |
| 4 | use super::hw::CanFrame; | import wiring; changing path/name can break compilation and module linkage. |
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceability. |
| 7 | use socketcan::{CanFrame as LinuxCanFrame, CanSocket, EmbeddedFrame, ExtendedId, Socket}; | import wiring; changin
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | #[derive(Debug, Default)] | comment/documentation; changing affects understanding and intent traceability. |
| 10 | pub struct SocketCanAdapter { | data layout contract; field changes can break callers/serde/tests. |
| 11 |     #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceability.
| 12 |     socket: Option<CanSocket>, | support logic; impact depends on neighboring block and data flow. |
| 13 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 14 | (blank) | blank separator; no runtime behavior. |
| 15 | impl SocketCanAdapter { | implementation binding; behavior semantics and trait conformance live here. |
| 16 |     #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceability.
| 17 |     pub fn open(iface: &str) -> Result<Self, HwError> { | function signature/body; changes affect API behavior a
| 18 |         let socket = CanSocket::open(iface).map_err(\|e\| HwError::PortNotFound(e.to_string()))?; | error propaga
| 19 |         socket | support logic; impact depends on neighboring block and data flow. |
| 20 |             .set_read_timeout(std::time::Duration::from_millis(250)) | support logic; impact depends on neighbor
| 21 |             .map_err(\|_e\| HwError::TransceiverError)?; | error propagation point; changes alter failure propaga
| 22 |         Ok(Self { | support logic; impact depends on neighboring block and data flow. |
| 23 |             socket: Some(socket), | support logic; impact depends on neighboring block and data flow. |
| 24 |         }) | support logic; impact depends on neighboring block and data flow. |
| 25 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 |     #[cfg(not(target_os = "linux"))] | comment/documentation; changing affects understanding and intent traceabil
| 28 |     pub fn open(_iface: &str) -> Result<Self, HwError> { | function signature/body; changes affect API behavior a
| 29 |         Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 30 |             "SocketCAN real adapter is supported on Linux only".to_string(), | support logic; impact depends on n
| 31 |         )) | support logic; impact depends on neighboring block and data flow. |
| 32 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 | impl HardwareInterface for SocketCanAdapter { | implementation binding; behavior semantics and trait conformance
| 36 |     fn init(&mut self, _config: &HwConfig) -> Result<(), HwError> { | function signature/body; changes affect API
| 37 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 38 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 | (blank) | blank separator; no runtime behavior. |
| 40 |     #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceability.
| 41 |     fn read_frame(&mut self) -> Result<Frame, HwError> { | function signature/body; changes affect API behavior a
| 42 |         let Some(socket) = self.socket.as_ref() else { | support logic; impact depends on neighboring block and d
| 43 |             return Err(HwError::PortNotFound("CAN socket not initialized".into())); | support logic; impact depen
| 44 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 45 | (blank) | blank separator; no runtime behavior. |
| 46 |         let f = socket.read_frame().map_err(\|e\| { | support logic; impact depends on neighboring block and data
| 47 |             if e.kind() == std::io::ErrorKind::TimedOut { | runtime gate; condition changes can enable/disable c
| 48 |                 HwError::Timeout | support logic; impact depends on neighboring block and data flow. |
| 49 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 50 |                 HwError::TransceiverError | support logic; impact depends on neighboring block and data flow. |
| 51 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 53 | (blank) | blank separator; no runtime behavior. |
| 54 |         let mut data = [0u8; 8]; | support logic; impact depends on neighboring block and data flow. |
```

```
| 55 |         let payload = f.data(); | support logic; impact depends on neighboring block and data flow. |
| 56 |         let len = payload.len().min(8); | support logic; impact depends on neighboring block and data flow. |
| 57 |         data[..len].copy_from_slice(&payload[..len]); | support logic; impact depends on neighboring block and da
| 58 | (blank) | blank separator; no runtime behavior. |
| 59 |         let id = match f.id() { | support logic; impact depends on neighboring block and data flow. |
| 60 |             socketcan::Id::Standard(stdid) => u32::from(stdid.as_raw()), | support logic; impact depends on neigh
| 61 |             socketcan::Id::Extended(extid) => extid.as_raw(), | support logic; impact depends on neighboring bloc
| 62 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 63 | (blank) | blank separator; no runtime behavior. |
| 64 |         Ok(Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 65 |             id, | support logic; impact depends on neighboring block and data flow. |
| 66 |             dlc: len as u8, | support logic; impact depends on neighboring block and data flow. |
| 67 |             data, | support logic; impact depends on neighboring block and data flow. |
| 68 |             len, | support logic; impact depends on neighboring block and data flow. |
| 69 |             timestamp_ms: Some(now_ms()), | support logic; impact depends on neighboring block and data flow. |
| 70 |         })) | support logic; impact depends on neighboring block and data flow. |
| 71 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 72 | (blank) | blank separator; no runtime behavior. |
| 73 |     #[cfg(not(target_os = "linux"))] | comment/documentation; changing affects understanding and intent traceabi
| 74 |     fn read_frame(&mut self) -> Result<Frame, HwError> { | function signature/body; changes affect API behavior a
| 75 |         Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 76 |             "SocketCAN real adapter is supported on Linux only".to_string(), | support logic; impact depends on n
| 77 |         )) | support logic; impact depends on neighboring block and data flow. |
| 78 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 79 | (blank) | blank separator; no runtime behavior. |
| 80 |     fn try_read_frame(&mut self) -> Result<Option<Frame>, HwError> { | function signature/body; changes affect AP
| 81 |         match self.read_frame() { | branch dispatcher; arm changes alter behavior class handling. |
| 82 |             Ok(f) => Ok(Some(f)), | support logic; impact depends on neighboring block and data flow. |
| 83 |             Err(HwError::Timeout) => Ok(None), | support logic; impact depends on neighboring block and data flow
| 84 |             Err(e) => Err(e), | support logic; impact depends on neighboring block and data flow. |
| 85 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 86 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 87 | (blank) | blank separator; no runtime behavior. |
| 88 |     #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceability.
| 89 |     fn send_frame(&mut self, frame: Frame) -> Result<(), HwError> { | function signature/body; changes affect API
| 90 |         let Some(socket) = self.socket.as_ref() else { | support logic; impact depends on neighboring block and d
| 91 |             return Err(HwError::PortNotFound("CAN socket not initialized".into())); | support logic; impact depen
| 92 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 93 | (blank) | blank separator; no runtime behavior. |
| 94 |         let cf = match frame { | support logic; impact depends on neighboring block and data flow. |
| 95 |             Frame::Can(c) => c, | support logic; impact depends on neighboring block and data flow. |
| 96 |             Frame::Serial(_) => { | support logic; impact depends on neighboring block and data flow. |
| 97 |                 return Err(HwError::ParseError { | support logic; impact depends on neighboring block and data fl
| 98 |                     cause: "cannot send serial frame over SocketCAN".to_string(), | protocol or I/O critical; cha
| 99 |                     raw_data: "serial-frame".to_string(), | support logic; impact depends on neighboring block an
| 100 |                 }) | support logic; impact depends on neighboring block and data flow. |
| 101 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 102 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 103 | (blank) | blank separator; no runtime behavior. |
| 104 |         let Some(ext_id) = ExtendedId::new(cf.id) else { | support logic; impact depends on neighboring block an
| 105 |             return Err(HwError::ParseError { | support logic; impact depends on neighboring block and data flow
| 106 |                 cause: "invalid CAN extended id".to_string(), | support logic; impact depends on neighboring blo
| 107 |                 raw_data: format!("{}", cf.id), | support logic; impact depends on neighboring block and data fl
| 108 |             }); | support logic; impact depends on neighboring block and data flow. |
| 109 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 110 | (blank) | blank separator; no runtime behavior. |
| 111 |         let Some(frame) = LinuxCanFrame::new(ext_id, &cf.data[..cf.len.min(8)]) else { | support logic; impact d
| 112 |             return Err(HwError::ParseError { | support logic; impact depends on neighboring block and data flow
| 113 |                 cause: "failed to build CAN frame".to_string(), | support logic; impact depends on neighboring b
| 114 |                 raw_data: format!("id={}", cf.id), | support logic; impact depends on neighboring block and data
| 115 |             }); | support logic; impact depends on neighboring block and data flow. |
| 116 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 117 | (blank) | blank separator; no runtime behavior. |
| 118 |         socket | support logic; impact depends on neighboring block and data flow. |
| 119 |             .write_frame(&frame) | protocol or I/O critical; changes may impact external integration correctness
| 120 |             .map_err(\|_e\| HwError::TransceiverError) | support logic; impact depends on neighboring block and
| 121 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 122 | (blank) | blank separator; no runtime behavior. |
| 123 |     #[cfg(not(target_os = "linux"))] | comment/documentation; changing affects understanding and intent traceab
| 124 |     fn send_frame(&mut self, _frame: Frame) -> Result<(), HwError> { | function signature/body; changes affect A
| 125 |         Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 126 |             "SocketCAN real adapter is supported on Linux only".to_string(), | support logic; impact depends on
| 127 |         )) | support logic; impact depends on neighboring block and data flow. |
| 128 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 129 | (blank) | blank separator; no runtime behavior. |
| 130 |     fn close(&mut self) -> Result<(), HwError> { | function signature/body; changes affect API behavior and con
```

```
| 131 |         #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceabi
| 132 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 133 |             self.socket = None; | support logic; impact depends on neighboring block and data flow. |
| 134 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 135 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 136 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 137 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 | (blank) | blank separator; no runtime behavior. |
| 139 | #[cfg(target_os = "linux")] | comment/documentation; changing affects understanding and intent traceability. |
| 140 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 141 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 142 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data flo
| 143 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 144 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 145 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/io/vendor_cat_comm.rs

Role: hardware/protocol integration, live flashing, diagnostics, and production gating

Line count: 418 | Non-empty: 377 | Symbol count: 27

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| send_bridge_request | 7 |
| map_bridge_error | 6 |
| new | 5 |
| supports | 4 |
| ensure_connected | 4 |
| bridge_info_from_response | 3 |
| is_empty | 3 |
| open | 2 |
| resolve_bridge_exe | 2 |
| write_effectively_enabled | 2 |
| ensure_bridge_compatibility | 2 |
| read_bridge_response | 2 |
| append_diag_record | 2 |
| now_ms | 2 |
| flush | 1 |
| init | 1 |
| read_frame | 1 |
| try_read_frame | 1 |
| send_frame | 1 |
| close | 1 |
| adapter_info | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 10 | const | DEFAULT_TEMPLATE_BRIDGE_EXE | namespace and compile-time linkage |
| 13 | struct | CatCommAdapter | data/schema coupling and match/serde break risk |
| 24 | struct | CatCommBridgeInfo | data/schema coupling and match/serde break risk |
| 29 | impl | CatCommBridgeInfo | method behavior and trait semantics |
| 30 | fn | supports | API and behavior coupling to callers/implementers |
| 37 | enum | BridgeRequest | data/schema coupling and match/serde break risk |
| 57 | struct | BridgeResponse | data/schema coupling and match/serde break risk |
| 68 | struct | BridgeDiagPayload | data/schema coupling and match/serde break risk |
| 77 | struct | BridgeDiagRecord | data/schema coupling and match/serde break risk |
| 87 | impl | CatCommAdapter | method behavior and trait semantics |
| 88 | fn | open | API and behavior coupling to callers/implementers |
| 154 | fn | ensure_connected | API and behavior coupling to callers/implementers |
| 164 | fn | send_bridge_request | API and behavior coupling to callers/implementers |
| 182 | fn | read_bridge_response | API and behavior coupling to callers/implementers |
| 201 | fn | append_diag_record | API and behavior coupling to callers/implementers |
| 232 | impl | HardwareInterface for CatCommAdapter | method behavior and trait semantics |
| 233 | fn | init | API and behavior coupling to callers/implementers |
| 246 | fn | read_frame | API and behavior coupling to callers/implementers |
| 261 | fn | try_read_frame | API and behavior coupling to callers/implementers |
| 273 | fn | send_frame | API and behavior coupling to callers/implementers |
| 286 | fn | close | API and behavior coupling to callers/implementers |
| 305 | fn | adapter_info | API and behavior coupling to callers/implementers |
```

```
| 318 | fn | resolve_bridge_exe | API and behavior coupling to callers/implementers |
| 347 | fn | map_bridge_error | API and behavior coupling to callers/implementers |
| 368 | fn | bridge_info_from_response | API and behavior coupling to callers/implementers |
| 375 | fn | ensure_bridge_compatibility | API and behavior coupling to callers/implementers |
| 413 | fn | now_ms | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use std::fs::{self, OpenOptions}; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | use std::io::{BufRead, BufReader, Write}; | import wiring; changing path/name can break compilation and module lin
| 3 | use std::path::PathBuf; | import wiring; changing path/name can break compilation and module linkage. |
| 4 | use std::process::{Child, ChildStdin, Command, Stdio}; | import wiring; changing path/name can break compilation a
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | use super::hw::{Frame, HardwareInterface, HwConfig, HwError}; | import wiring; changing path/name can break compil
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | const DEFAULT_TEMPLATE_BRIDGE_EXE: &str = "cat_comm_bridge.exe"; | support logic; impact depends on neighboring b
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | #[derive(Debug)] | comment/documentation; changing affects understanding and intent traceability. |
| 13 | pub struct CatCommAdapter { | data layout contract; field changes can break callers/serde/tests. |
| 14 |     connected: bool, | support logic; impact depends on neighboring block and data flow. |
| 15 |     child: Child, | support logic; impact depends on neighboring block and data flow. |
| 16 |     child_stdin: ChildStdin, | support logic; impact depends on neighboring block and data flow. |
| 17 |     child_stdout: BufReader<std::process::ChildStdout>, | support logic; impact depends on neighboring block and
| 18 |     timeout_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 19 |     bridge_info: CatCommBridgeInfo, | support logic; impact depends on neighboring block and data flow. |
| 20 |     diag_log_path: PathBuf, | support logic; impact depends on neighboring block and data flow. |
| 21 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 22 | (blank) | blank separator; no runtime behavior. |
| 23 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 24 | struct CatCommBridgeInfo { | data layout contract; field changes can break callers/serde/tests. |
| 25 |     protocol_version: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 26 |     capabilities: Vec<String>, | support logic; impact depends on neighboring block and data flow. |
| 27 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 28 | (blank) | blank separator; no runtime behavior. |
| 29 | impl CatCommBridgeInfo { | implementation binding; behavior semantics and trait conformance live here. |
| 30 |     fn supports(&self, cap: &str) -> bool { | function signature/body; changes affect API behavior and control-f
| 31 |         self.capabilities.iter().any(\|c\| c.eq_ignore_ascii_case(cap)) | support logic; impact depends on neighb
| 32 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 | #[derive(Debug, Serialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 36 | #[serde(tag = "cmd", rename_all = "snake_case")] | comment/documentation; changing affects understanding and inte
| 37 | enum BridgeRequest { | state machine variants; changes ripple through matches and business logic. |
| 38 |     Init { | support logic; impact depends on neighboring block and data flow. |
| 39 |         dry_run: bool, | support logic; impact depends on neighboring block and data flow. |
| 40 |         enable_write: bool, | protocol or I/O critical; changes may impact external integration correctness. |
| 41 |         can_interface: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 42 |         serial_port: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 43 |         template_dir: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 44 |     }, | support logic; impact depends on neighboring block and data flow. |
| 45 |     Read { | support logic; impact depends on neighboring block and data flow. |
| 46 |         timeout_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 47 |     }, | support logic; impact depends on neighboring block and data flow. |
| 48 |     TryRead, | support logic; impact depends on neighboring block and data flow. |
| 49 |     Send { | support logic; impact depends on neighboring block and data flow. |
| 50 |         frame: Frame, | support logic; impact depends on neighboring block and data flow. |
| 51 |     }, | support logic; impact depends on neighboring block and data flow. |
| 52 |     Close, | support logic; impact depends on neighboring block and data flow. |
| 53 |     Ping, | support logic; impact depends on neighboring block and data flow. |
| 54 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 | (blank) | blank separator; no runtime behavior. |
| 56 | #[derive(Debug, Deserialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 57 | struct BridgeResponse { | data layout contract; field changes can break callers/serde/tests. |
| 58 |     ok: bool, | support logic; impact depends on neighboring block and data flow. |
| 59 |     frame: Option<Frame>, | support logic; impact depends on neighboring block and data flow. |
| 60 |     error: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 61 |     kind: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 62 |     protocol_version: Option<String>, | support logic; impact depends on neighboring block and data flow. |
| 63 |     capabilities: Option<Vec<String>>, | support logic; impact depends on neighboring block and data flow. |
| 64 |     diag: Option<BridgeDiagPayload>, | support logic; impact depends on neighboring block and data flow. |
| 65 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 66 | (blank) | blank separator; no runtime behavior. |
```

```
| 67 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
| 68 | struct BridgeDiagPayload { | data layout contract; field changes can break callers/serde/tests. |
| 69 |     wait_frame_count_delta: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 70 |     sequence_error_count_delta: Option<u32>, | support logic; impact depends on neighboring block and data flow.
| 71 |     flowcontrol_timeout_count_delta: Option<u32>, | support logic; impact depends on neighboring block and data f
| 72 |     fc_blocksize: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 73 |     fc_stmin_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
| 74 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 75 | (blank) | blank separator; no runtime behavior. |
| 76 | #[derive(Debug, Serialize)] | comment/documentation; changing affects understanding and intent traceability. |
| 77 | struct BridgeDiagRecord<'a> { | data layout contract; field changes can break callers/serde/tests. |
| 78 |     ts_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 79 |     source: &'a str, | support logic; impact depends on neighboring block and data flow. |
| 80 |     wait_frame_count_delta: u32, | support logic; impact depends on neighboring block and data flow. |
| 81 |     sequence_error_count_delta: u32, | support logic; impact depends on neighboring block and data flow. |
| 82 |     flowcontrol_timeout_count_delta: u32, | support logic; impact depends on neighboring block and data flow. |
| 83 |     fc_blocksize: Option<u8>, | support logic; impact depends on neighboring block and data flow. |
| 84 |     fc_stmin_ms: Option<u64>, | support logic; impact depends on neighboring block and data flow. |
| 85 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 86 | (blank) | blank separator; no runtime behavior. |
| 87 | impl CatCommAdapter { | implementation binding; behavior semantics and trait conformance live here. |
| 88 |     pub fn open(cfg: &HwConfig) -> Result<Self, HwError> { | function signature/body; changes affect API behavior
| 89 |         let _vendor = cfg | support logic; impact depends on neighboring block and data flow. |
| 90 |             .vendor_name | support logic; impact depends on neighboring block and data flow. |
| 91 |             .as_deref() | support logic; impact depends on neighboring block and data flow. |
| 92 |             .unwrap_or("cat_comm") | support logic; impact depends on neighboring block and data flow. |
| 93 |             .to_ascii_lowercase(); | support logic; impact depends on neighboring block and data flow. |
| 94 | (blank) | blank separator; no runtime behavior. |
| 95 |         let bridge_exe = resolve_bridge_exe(cfg)?; | error propagation point; changes alter failure propagation s
| 96 |         let mut child = Command::new(&bridge_exe) | support logic; impact depends on neighboring block and data f
| 97 |             .arg("--mode=stdio-jsonl") | support logic; impact depends on neighboring block and data flow. |
| 98 |             .stdin(Stdio::piped()) | support logic; impact depends on neighboring block and data flow. |
| 99 |             .stdout(Stdio::piped()) | support logic; impact depends on neighboring block and data flow. |
| 100 |             .stderr(Stdio::piped()) | support logic; impact depends on neighboring block and data flow. |
| 101 |             .spawn() | support logic; impact depends on neighboring block and data flow. |
| 102 |             .map_err(\|e\| { | support logic; impact depends on neighboring block and data flow. |
| 103 |                 HwError::Unknown(format!( | support logic; impact depends on neighboring block and data flow. |
| 104 |                     "failed to start Cat Comm bridge at {}: {e}", | support logic; impact depends on neighboring
| 105 |                     bridge_exe.display() | support logic; impact depends on neighboring block and data flow. |
| 106 |                 )) | support logic; impact depends on neighboring block and data flow. |
| 107 |             })?; | error propagation point; changes alter failure propagation semantics. |
| 108 | (blank) | blank separator; no runtime behavior. |
| 109 |         let child_stdin = child.stdin.take().ok_or_else(\|\| { | support logic; impact depends on neighboring b
| 110 |             HwError::Unknown("failed to capture Cat Comm bridge stdin".to_string()) | support logic; impact depe
| 111 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 112 |         let child_stdout = child.stdout.take().ok_or_else(\|\| { | support logic; impact depends on neighboring
| 113 |             HwError::Unknown("failed to capture Cat Comm bridge stdout".to_string()) | support logic; impact de
| 114 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 115 | (blank) | blank separator; no runtime behavior. |
| 116 |         let mut adapter = Self { | support logic; impact depends on neighboring block and data flow. |
| 117 |             connected: false, | support logic; impact depends on neighboring block and data flow. |
| 118 |             child, | support logic; impact depends on neighboring block and data flow. |
| 119 |             child_stdin, | support logic; impact depends on neighboring block and data flow. |
| 120 |             child_stdout: BufReader::new(child_stdout), | support logic; impact depends on neighboring block and
| 121 |             timeout_ms: cfg.vendor_bridge_timeout_ms.max(100), | support logic; impact depends on neighboring b
| 122 |             bridge_info: CatCommBridgeInfo { | support logic; impact depends on neighboring block and data flow
| 123 |                 protocol_version: None, | support logic; impact depends on neighboring block and data flow. |
| 124 |                 capabilities: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 125 |             }, | support logic; impact depends on neighboring block and data flow. |
| 126 |             diag_log_path: cfg.log_dir.join("cat_comm_bridge_diag.jsonl"), | support logic; impact depends on n
| 127 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 128 |         fs::create_dir_all(&cfg.log_dir) | support logic; impact depends on neighboring block and data flow. |
| 129 |             .map_err(\|e\| HwError::Unknown(format!("create bridge log dir failed: {e}")))?; | error propagatio
| 130 | (blank) | blank separator; no runtime behavior. |
| 131 |         let init_req = BridgeRequest::Init { | support logic; impact depends on neighboring block and data flow
| 132 |             dry_run: cfg.dry_run, | support logic; impact depends on neighboring block and data flow. |
| 133 |             enable_write: cfg.write_effectively_enabled(), | protocol or I/O critical; changes may impact extern
| 134 |             can_interface: cfg.can_interface.clone(), | support logic; impact depends on neighboring block and
| 135 |             serial_port: cfg.serial_port.clone(), | support logic; impact depends on neighboring block and data
| 136 |             template_dir: cfg | support logic; impact depends on neighboring block and data flow. |
| 137 |                 .vendor_template_dir | support logic; impact depends on neighboring block and data flow. |
| 138 |                 .as_ref() | support logic; impact depends on neighboring block and data flow. |
| 139 |                 .map(\|p\| p.display().to_string()), | support logic; impact depends on neighboring block and da
| 140 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 141 |         let init_rsp = adapter.send_bridge_request(&init_req)?; | error propagation point; changes alter failure
| 142 |         if !init_rsp.ok { | runtime gate; condition changes can enable/disable critical paths. |
```

```
| 143 |             return Err(map_bridge_error( | support logic; impact depends on neighboring block and data flow. |
| 144 |                 init_rsp.kind, | support logic; impact depends on neighboring block and data flow. |
| 145 |                 init_rsp.error.unwrap_or_else(\|\| "bridge init failed".to_string()), | support logic; impact de
| 146 |             )); | support logic; impact depends on neighboring block and data flow. |
| 147 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 148 |         adapter.bridge_info = bridge_info_from_response(&init_rsp); | support logic; impact depends on neighbori
| 149 |         ensure_bridge_compatibility(cfg, &adapter.bridge_info)?; | error propagation point; changes alter failu
| 150 | (blank) | blank separator; no runtime behavior. |
| 151 |         Ok(adapter) | support logic; impact depends on neighboring block and data flow. |
| 152 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 153 | (blank) | blank separator; no runtime behavior. |
| 154 |     fn ensure_connected(&self) -> Result<(), HwError> { | function signature/body; changes affect API behavior a
| 155 |         if self.connected { | runtime gate; condition changes can enable/disable critical paths. |
| 156 |             Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 157 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 158 |             Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 159 |                 "Cat Comm bridge is not initialized".to_string(), | support logic; impact depends on neighboring
| 160 |             )) | support logic; impact depends on neighboring block and data flow. |
| 161 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 162 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 163 | (blank) | blank separator; no runtime behavior. |
| 164 |     fn send_bridge_request(&mut self, req: &BridgeRequest) -> Result<BridgeResponse, HwError> { | function signa
| 165 |         let line = serde_json::to_string(req) | support logic; impact depends on neighboring block and data flow
| 166 |             .map_err(\|e\| HwError::Unknown(format!("bridge request encode failed: {e}")))?; | error propagatio
| 167 |         self.child_stdin | support logic; impact depends on neighboring block and data flow. |
| 168 |             .write_all(line.as_bytes()) | protocol or I/O critical; changes may impact external integration cor
| 169 |             .map_err(\|e\| HwError::Unknown(format!("bridge stdin write failed: {e}")))?; | error propagation p
| 170 |         self.child_stdin | support logic; impact depends on neighboring block and data flow. |
| 171 |             .write_all(b"\n") | protocol or I/O critical; changes may impact external integration correctness.
| 172 |             .map_err(\|e\| HwError::Unknown(format!("bridge stdin newline failed: {e}")))?; | error propagation
| 173 |         self.child_stdin | support logic; impact depends on neighboring block and data flow. |
| 174 |             .flush() | support logic; impact depends on neighboring block and data flow. |
| 175 |             .map_err(\|e\| HwError::Unknown(format!("bridge stdin flush failed: {e}")))?; | error propagation p
| 176 | (blank) | blank separator; no runtime behavior. |
| 177 |         let rsp = self.read_bridge_response()?; | error propagation point; changes alter failure propagation se
| 178 |         self.append_diag_record(&rsp); | support logic; impact depends on neighboring block and data flow. |
| 179 |         Ok(rsp) | support logic; impact depends on neighboring block and data flow. |
| 180 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 181 | (blank) | blank separator; no runtime behavior. |
| 182 |     fn read_bridge_response(&mut self) -> Result<BridgeResponse, HwError> { | function signature/body; changes a
| 183 |         let mut buf = String::new(); | support logic; impact depends on neighboring block and data flow. |
| 184 |         self.child_stdout | support logic; impact depends on neighboring block and data flow. |
| 185 |             .read_line(&mut buf) | support logic; impact depends on neighboring block and data flow. |
| 186 |             .map_err(\|e\| HwError::Unknown(format!("bridge stdout read failed: {e}")))?; | error propagation p
| 187 |         if buf.trim().is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 188 |             return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 189 |                 "Cat Comm bridge closed stream or returned empty response".to_string(), | support logic; impact
| 190 |             )); | support logic; impact depends on neighboring block and data flow. |
| 191 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 192 | (blank) | blank separator; no runtime behavior. |
| 193 |         serde_json::from_str::<BridgeResponse>(buf.trim()).map_err(\|e\| { | support logic; impact depends on n
| 194 |             HwError::ParseError { | support logic; impact depends on neighboring block and data flow. |
| 195 |                 cause: format!("bridge response parse failed: {e}"), | support logic; impact depends on neighbo
| 196 |                 raw_data: buf, | support logic; impact depends on neighboring block and data flow. |
| 197 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 198 |         }) | support logic; impact depends on neighboring block and data flow. |
| 199 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 200 | (blank) | blank separator; no runtime behavior. |
| 201 |     fn append_diag_record(&self, rsp: &BridgeResponse) { | function signature/body; changes affect API behavior
| 202 |         let Some(diag) = &rsp.diag else { | support logic; impact depends on neighboring block and data flow. |
| 203 |             return; | support logic; impact depends on neighboring block and data flow. |
| 204 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 205 | (blank) | blank separator; no runtime behavior. |
| 206 |         let rec = BridgeDiagRecord { | support logic; impact depends on neighboring block and data flow. |
| 207 |             ts_ms: now_ms(), | support logic; impact depends on neighboring block and data flow. |
| 208 |             source: "vendor_cat_comm", | support logic; impact depends on neighboring block and data flow. |
| 209 |             wait_frame_count_delta: diag.wait_frame_count_delta.unwrap_or(0), | support logic; impact depends o
| 210 |             sequence_error_count_delta: diag.sequence_error_count_delta.unwrap_or(0), | support logic; impact de
| 211 |             flowcontrol_timeout_count_delta: diag.flowcontrol_timeout_count_delta.unwrap_or(0), | support logic
| 212 |             fc_blocksize: diag.fc_blocksize, | support logic; impact depends on neighboring block and data flow
| 213 |             fc_stmin_ms: diag.fc_stmin_ms, | support logic; impact depends on neighboring block and data flow.
| 214 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 215 | (blank) | blank separator; no runtime behavior. |
| 216 |         let Ok(mut f) = OpenOptions::new() | support logic; impact depends on neighboring block and data flow.
| 217 |             .create(true) | support logic; impact depends on neighboring block and data flow. |
| 218 |             .append(true) | support logic; impact depends on neighboring block and data flow. |
```

```
| 219 |             .open(&self.diag_log_path) | support logic; impact depends on neighboring block and data flow. |
| 220 |         else { | support logic; impact depends on neighboring block and data flow. |
| 221 |             return; | support logic; impact depends on neighboring block and data flow. |
| 222 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 223 | (blank) | blank separator; no runtime behavior. |
| 224 |         let line = match serde_json::to_string(&rec) { | support logic; impact depends on neighboring block and
| 225 |             Ok(v) => v, | support logic; impact depends on neighboring block and data flow. |
| 226 |             Err(_) => return, | support logic; impact depends on neighboring block and data flow. |
| 227 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 228 |         let _ = writeln!(f, "{line}"); | protocol or I/O critical; changes may impact external integration corr
| 229 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 230 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 231 | (blank) | blank separator; no runtime behavior. |
| 232 | impl HardwareInterface for CatCommAdapter { | implementation binding; behavior semantics and trait conformance
| 233 |     fn init(&mut self, _config: &HwConfig) -> Result<(), HwError> { | function signature/body; changes affect A
| 234 |         let rsp = self.send_bridge_request(&BridgeRequest::Ping)?; | error propagation point; changes alter fai
| 235 |         if rsp.ok { | runtime gate; condition changes can enable/disable critical paths. |
| 236 |             self.bridge_info = bridge_info_from_response(&rsp); | support logic; impact depends on neighboring
| 237 |             self.connected = true; | support logic; impact depends on neighboring block and data flow. |
| 238 |             return Ok(()); | support logic; impact depends on neighboring block and data flow. |
| 239 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 240 |         Err(map_bridge_error( | support logic; impact depends on neighboring block and data flow. |
| 241 |             rsp.kind, | support logic; impact depends on neighboring block and data flow. |
| 242 |             rsp.error.unwrap_or_else(\|\| "bridge ping failed".to_string()), | support logic; impact depends on
| 243 |         )) | support logic; impact depends on neighboring block and data flow. |
| 244 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 245 | (blank) | blank separator; no runtime behavior. |
| 246 |     fn read_frame(&mut self) -> Result<Frame, HwError> { | function signature/body; changes affect API behavior
| 247 |         self.ensure_connected()?; | error propagation point; changes alter failure propagation semantics. |
| 248 |         let rsp = self.send_bridge_request(&BridgeRequest::Read { | protocol or I/O critical; changes may impac
| 249 |             timeout_ms: self.timeout_ms, | support logic; impact depends on neighboring block and data flow. |
| 250 |         })?; | error propagation point; changes alter failure propagation semantics. |
| 251 |         if !rsp.ok { | runtime gate; condition changes can enable/disable critical paths. |
| 252 |             return Err(map_bridge_error( | support logic; impact depends on neighboring block and data flow. |
| 253 |                 rsp.kind, | support logic; impact depends on neighboring block and data flow. |
| 254 |                 rsp.error.unwrap_or_else(\|\| "bridge read failed".to_string()), | support logic; impact depend
| 255 |             )); | support logic; impact depends on neighboring block and data flow. |
| 256 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 257 |         rsp.frame | support logic; impact depends on neighboring block and data flow. |
| 258 |             .ok_or_else(\|\| HwError::Unknown("bridge read returned no frame".to_string())) | support logic; im
| 259 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 260 | (blank) | blank separator; no runtime behavior. |
| 261 |     fn try_read_frame(&mut self) -> Result<Option<Frame>, HwError> { | function signature/body; changes affect
| 262 |         self.ensure_connected()?; | error propagation point; changes alter failure propagation semantics. |
| 263 |         let rsp = self.send_bridge_request(&BridgeRequest::TryRead)?; | error propagation point; changes alter
| 264 |         if !rsp.ok { | runtime gate; condition changes can enable/disable critical paths. |
| 265 |             return Err(map_bridge_error( | support logic; impact depends on neighboring block and data flow. |
| 266 |                 rsp.kind, | support logic; impact depends on neighboring block and data flow. |
| 267 |                 rsp.error.unwrap_or_else(\|\| "bridge try_read failed".to_string()), | support logic; impact de
| 268 |             )); | support logic; impact depends on neighboring block and data flow. |
| 269 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 270 |         Ok(rsp.frame) | support logic; impact depends on neighboring block and data flow. |
| 271 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 272 | (blank) | blank separator; no runtime behavior. |
| 273 |     fn send_frame(&mut self, frame: Frame) -> Result<(), HwError> { | function signature/body; changes affect A
| 274 |         self.ensure_connected()?; | error propagation point; changes alter failure propagation semantics. |
| 275 |         let rsp = self.send_bridge_request(&BridgeRequest::Send { frame })?; | error propagation point; changes
| 276 |         if rsp.ok { | runtime gate; condition changes can enable/disable critical paths. |
| 277 |             Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 278 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 279 |             Err(map_bridge_error( | support logic; impact depends on neighboring block and data flow. |
| 280 |                 rsp.kind, | support logic; impact depends on neighboring block and data flow. |
| 281 |                 rsp.error.unwrap_or_else(\|\| "bridge send failed".to_string()), | protocol or I/O critical; ch
| 282 |             )) | support logic; impact depends on neighboring block and data flow. |
| 283 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 284 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 285 | (blank) | blank separator; no runtime behavior. |
| 286 |     fn close(&mut self) -> Result<(), HwError> { | function signature/body; changes affect API behavior and con
| 287 |         let _ = self.send_bridge_request(&BridgeRequest::Close); | protocol or I/O critical; changes may impact
| 288 |         self.connected = false; | support logic; impact depends on neighboring block and data flow. |
| 289 | (blank) | blank separator; no runtime behavior. |
| 290 |         if let Err(e) = self.child.kill() { | runtime gate; condition changes can enable/disable critical paths
| 291 |             let already_exited = self | support logic; impact depends on neighboring block and data flow. |
| 292 |                 .child | support logic; impact depends on neighboring block and data flow. |
| 293 |                 .try_wait() | support logic; impact depends on neighboring block and data flow. |
| 294 |                 .ok() | support logic; impact depends on neighboring block and data flow. |
```

```
| 295 |                 .flatten() | support logic; impact depends on neighboring block and data flow. |
| 296 |                 .is_some(); | support logic; impact depends on neighboring block and data flow. |
| 297 |             if !already_exited { | runtime gate; condition changes can enable/disable critical paths. |
| 298 |                 return Err(HwError::Unknown(format!("failed to stop Cat Comm bridge: {e}"))); | support logic; i
| 299 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 300 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 301 |         let _ = self.child.wait(); | support logic; impact depends on neighboring block and data flow. |
| 302 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 303 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 304 | (blank) | blank separator; no runtime behavior. |
| 305 |     fn adapter_info(&self) -> Option<String> { | function signature/body; changes affect API behavior and contro
| 306 |         Some(format!( | support logic; impact depends on neighboring block and data flow. |
| 307 |             "bridge_protocol={:?} capabilities={}", | error propagation point; changes alter failure propagation
| 308 |             self.bridge_info.protocol_version, | support logic; impact depends on neighboring block and data flo
| 309 |             if self.bridge_info.capabilities.is_empty() { | runtime gate; condition changes can enable/disable c
| 310 |                 "none".to_string() | support logic; impact depends on neighboring block and data flow. |
| 311 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 312 |                 self.bridge_info.capabilities.join(",") | support logic; impact depends on neighboring block and
| 313 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 314 |         )) | support logic; impact depends on neighboring block and data flow. |
| 315 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 316 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 317 | (blank) | blank separator; no runtime behavior. |
| 318 | fn resolve_bridge_exe(cfg: &HwConfig) -> Result<PathBuf, HwError> { | function signature/body; changes affect AP
| 319 |     if let Some(explicit) = &cfg.vendor_bridge_exe { | runtime gate; condition changes can enable/disable critic
| 320 |         if explicit.exists() { | runtime gate; condition changes can enable/disable critical paths. |
| 321 |             return Ok(explicit.clone()); | support logic; impact depends on neighboring block and data flow. |
| 322 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 323 |         return Err(HwError::PortNotFound(format!( | support logic; impact depends on neighboring block and data
| 324 |             "vendor bridge executable not found: {}", | support logic; impact depends on neighboring block and c
| 325 |             explicit.display() | support logic; impact depends on neighboring block and data flow. |
| 326 |         ))); | support logic; impact depends on neighboring block and data flow. |
| 327 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 328 | (blank) | blank separator; no runtime behavior. |
| 329 |     if let Some(template_dir) = &cfg.vendor_template_dir { | runtime gate; condition changes can enable/disable
| 330 |         let candidate = template_dir.join(DEFAULT_TEMPLATE_BRIDGE_EXE); | support logic; impact depends on neigh
| 331 |         if candidate.exists() { | runtime gate; condition changes can enable/disable critical paths. |
| 332 |             return Ok(candidate); | support logic; impact depends on neighboring block and data flow. |
| 333 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 334 |         return Err(HwError::PortNotFound(format!( | support logic; impact depends on neighboring block and data
| 335 |             "template bridge not found at {}. Upload your template bridge executable as {}", | support logic; im
| 336 |             candidate.display(), | support logic; impact depends on neighboring block and data flow. |
| 337 |             DEFAULT_TEMPLATE_BRIDGE_EXE | support logic; impact depends on neighboring block and data flow. |
| 338 |         ))); | support logic; impact depends on neighboring block and data flow. |
| 339 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 | (blank) | blank separator; no runtime behavior. |
| 341 |     Err(HwError::PortNotFound( | support logic; impact depends on neighboring block and data flow. |
| 342 |         "Cat Comm bridge not configured. Set --vendor-bridge-exe=<path> or --vendor-template-dir=<dir>" | suppo
| 343 |             .to_string(), | support logic; impact depends on neighboring block and data flow. |
| 344 |     )) | support logic; impact depends on neighboring block and data flow. |
| 345 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 346 | (blank) | blank separator; no runtime behavior. |
| 347 | fn map_bridge_error(kind: Option<String>, msg: String) -> HwError { | function signature/body; changes affect AP
| 348 |     match kind | branch dispatcher; arm changes alter behavior class handling. |
| 349 |         .as_deref() | support logic; impact depends on neighboring block and data flow. |
| 350 |         .unwrap_or_default() | support logic; impact depends on neighboring block and data flow. |
| 351 |         .to_ascii_lowercase() | support logic; impact depends on neighboring block and data flow. |
| 352 |         .as_str() | support logic; impact depends on neighboring block and data flow. |
| 353 |     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 354 |         "timeout" => HwError::Timeout, | support logic; impact depends on neighboring block and data flow. |
| 355 |         "rate_limited" => HwError::RateLimited, | support logic; impact depends on neighboring block and data f
| 356 |         "bus_off" => HwError::BusOff, | support logic; impact depends on neighboring block and data flow. |
| 357 |         "transceiver" => HwError::TransceiverError, | support logic; impact depends on neighboring block and dat
| 358 |         "permission" => HwError::PermissionDenied(msg), | support logic; impact depends on neighboring block and
| 359 |         "write_blocked" => HwError::WriteBlockedAllowlist, | protocol or I/O critical; changes may impact extern
| 360 |         "parse" => HwError::ParseError { | support logic; impact depends on neighboring block and data flow. |
| 361 |             cause: "bridge parse error".to_string(), | support logic; impact depends on neighboring block and da
| 362 |             raw_data: msg, | support logic; impact depends on neighboring block and data flow. |
| 363 |         }, | support logic; impact depends on neighboring block and data flow. |
| 364 |         _ => HwError::Unknown(msg), | support logic; impact depends on neighboring block and data flow. |
| 365 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 366 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 367 | (blank) | blank separator; no runtime behavior. |
| 368 | fn bridge_info_from_response(rsp: &BridgeResponse) -> CatCommBridgeInfo { | function signature/body; changes af
| 369 |     CatCommBridgeInfo { | support logic; impact depends on neighboring block and data flow. |
| 370 |         protocol_version: rsp.protocol_version.clone(), | support logic; impact depends on neighboring block and
```

```
| 371 |         capabilities: rsp.capabilities.clone().unwrap_or_default(), | support logic; impact depends on neighbori
| 372 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 373 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 374 | (blank) | blank separator; no runtime behavior. |
| 375 | fn ensure_bridge_compatibility(cfg: &HwConfig, info: &CatCommBridgeInfo) -> Result<(), HwError> { | function sig
| 376 |     if let Some(ver) = &info.protocol_version { | runtime gate; condition changes can enable/disable critical pa
| 377 |         let major = ver | support logic; impact depends on neighboring block and data flow. |
| 378 |             .split('.') | support logic; impact depends on neighboring block and data flow. |
| 379 |             .next() | support logic; impact depends on neighboring block and data flow. |
| 380 |             .and_then(\|s\| s.parse::<u32>().ok()) | support logic; impact depends on neighboring block and data
| 381 |             .unwrap_or(0); | support logic; impact depends on neighboring block and data flow. |
| 382 |         if major > 1 { | runtime gate; condition changes can enable/disable critical paths. |
| 383 |             return Err(HwError::Unknown(format!( | support logic; impact depends on neighboring block and data f
| 384 |                 "unsupported Cat Comm bridge protocol version {ver}; expected major version 1" | support logic;
| 385 |             ))); | support logic; impact depends on neighboring block and data flow. |
| 386 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 387 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 388 | (blank) | blank separator; no runtime behavior. |
| 389 |     // Capability checks are strict only when the bridge explicitly reports capabilities. | comment/documentatio
| 390 |     if !info.capabilities.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 391 |         if cfg.write_effectively_enabled() && !info.supports("write") { | runtime gate; condition changes can en
| 392 |             return Err(HwError::PermissionDenied( | support logic; impact depends on neighboring block and data
| 393 |                 "bridge does not advertise write capability".to_string(), | protocol or I/O critical; changes ma
| 394 |             )); | support logic; impact depends on neighboring block and data flow. |
| 395 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 396 |         if cfg.can_interface.is_some() && !info.supports("can") { | runtime gate; condition changes can enable/d
| 397 |             return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 398 |                 "bridge does not advertise CAN capability for requested --can-if path" | support logic; impact c
| 399 |                     .to_string(), | support logic; impact depends on neighboring block and data flow. |
| 400 |             )); | support logic; impact depends on neighboring block and data flow. |
| 401 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 402 |         if cfg.serial_port.is_some() && !info.supports("serial") { | runtime gate; condition changes can enable/
| 403 |             return Err(HwError::Unknown( | support logic; impact depends on neighboring block and data flow. |
| 404 |                 "bridge does not advertise serial capability for requested --serial-port path" | support logic;
| 405 |                     .to_string(), | support logic; impact depends on neighboring block and data flow. |
| 406 |             )); | support logic; impact depends on neighboring block and data flow. |
| 407 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 408 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 409 | (blank) | blank separator; no runtime behavior. |
| 410 |     Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 411 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 412 | (blank) | blank separator; no runtime behavior. |
| 413 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 414 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 415 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data flo
| 416 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 417 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 418 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/j1939.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 1656 | Non-empty: 1611 | Symbol count: 129

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| from_raw | 17 |
| build_id | 17 |
| pgn_name | 5 |
| new | 4 |
| find_pgn | 3 |
| len | 3 |
| decode | 2 |
| bits | 2 |
| push | 2 |
| adas1 | 2 |
| engy1 | 2 |
| sa_name | 1 |
| data_hex | 1 |
| eec1 | 1 |
| eec2 | 1 |
```

| et1 | 1 |
| efl_p1 | 1 |
| ic1 | 1 |
| lfe | 1 |
| pto | 1 |
| ccvs | 1 |
| dm1 | 1 |
| hours | 1 |
| fus1 | 1 |
| fmt | 1 |

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 14 | module | addr | namespace and compile-time linkage |
| 15 | const | ECM_1 | namespace and compile-time linkage |
| 16 | const | ECM_2 | namespace and compile-time linkage |
| 17 | const | TURBOCHARGER | namespace and compile-time linkage |
| 18 | const | TRANSMISSION | namespace and compile-time linkage |
| 19 | const | BRAKES | namespace and compile-time linkage |
| 20 | const | TASK_CTRL | namespace and compile-time linkage |
| 21 | const | HEADWAY | namespace and compile-time linkage |
| 22 | const | IMPLEMENT | namespace and compile-time linkage |
| 23 | const | ISOBUS_VT | namespace and compile-time linkage |
| 24 | const | CAB | namespace and compile-time linkage |
| 25 | const | ARMREST | namespace and compile-time linkage |
| 26 | const | INSTRUMENT | namespace and compile-time linkage |
| 27 | const | HITCH | namespace and compile-time linkage |
| 28 | const | NULL | namespace and compile-time linkage |
| 29 | const | BROADCAST | namespace and compile-time linkage |
| 33 | module | pgn | namespace and compile-time linkage |
| 35 | const | EBC1 | namespace and compile-time linkage |
| 36 | const | ETC1 | namespace and compile-time linkage |
| 37 | const | ETC2 | namespace and compile-time linkage |
| 38 | const | EEC1 | namespace and compile-time linkage |
| 39 | const | EEC2 | namespace and compile-time linkage |
| 40 | const | EEC3 | namespace and compile-time linkage |
| 41 | const | ET1 | namespace and compile-time linkage |
| 42 | const | EFL_P1 | namespace and compile-time linkage |
| 43 | const | IC1 | namespace and compile-time linkage |
| 44 | const | LFE | namespace and compile-time linkage |
| 45 | const | HOURS | namespace and compile-time linkage |
| 46 | const | FUEL1 | namespace and compile-time linkage |
| 47 | const | EEC5 | namespace and compile-time linkage |
| 49 | const | CCVS | namespace and compile-time linkage |
| 50 | const | VD | namespace and compile-time linkage |
| 51 | const | AMB | namespace and compile-time linkage |
| 52 | const | FD | namespace and compile-time linkage |
| 54 | const | PTO | namespace and compile-time linkage |
| 55 | const | HITCH | namespace and compile-time linkage |
| 56 | const | WS_MST | namespace and compile-time linkage |
| 57 | const | WS_MBR | namespace and compile-time linkage |
| 58 | const | SC | namespace and compile-time linkage |
| 59 | const | GSC | namespace and compile-time linkage |
| 61 | const | DM1 | namespace and compile-time linkage |
| 62 | const | DM2 | namespace and compile-time linkage |
| 63 | const | DM3 | namespace and compile-time linkage |
| 64 | const | DM11 | namespace and compile-time linkage |
| 65 | const | DM15 | namespace and compile-time linkage |
| 66 | const | DM16 | namespace and compile-time linkage |
| 67 | const | RQST | namespace and compile-time linkage |
| 69 | const | TP_CM | namespace and compile-time linkage |
| 70 | const | TP_DT | namespace and compile-time linkage |
| 71 | const | ECAN | namespace and compile-time linkage |
| 72 | const | PROP_A | namespace and compile-time linkage |
| 73 | const | ADAS1 | namespace and compile-time linkage |
| 74 | const | FUS1 | namespace and compile-time linkage |
| 75 | const | ENGY1 | namespace and compile-time linkage |
| 78 | const | TSC1 | namespace and compile-time linkage |
| 79 | const | CM1 | namespace and compile-time linkage |
| 80 | const | TCFG | namespace and compile-time linkage |
| 81 | const | EC1 | namespace and compile-time linkage |
| 82 | const | HRVD | namespace and compile-time linkage |
| 83 | const | SHUTDN | namespace and compile-time linkage |
| 84 | const | SOFT | namespace and compile-time linkage |

| 85 | const | ECFG | namespace and compile-time linkage |
| 86 | const | PTODE | namespace and compile-time linkage |
| 87 | const | TD | namespace and compile-time linkage |
| 88 | const | TIRE | namespace and compile-time linkage |
| 89 | const | EI | namespace and compile-time linkage |
| 90 | const | VEP1 | namespace and compile-time linkage |
| 91 | const | SERV | namespace and compile-time linkage |
| 92 | const | EBC2 | namespace and compile-time linkage |
| 94 | const | GSC1 | namespace and compile-time linkage |
| 96 | const | NM_ACK | namespace and compile-time linkage |
| 97 | const | DM13 | namespace and compile-time linkage |
| 98 | const | DM14 | namespace and compile-time linkage |
| 99 | const | DM18 | namespace and compile-time linkage |
| 100 | const | DM19 | namespace and compile-time linkage |
| 101 | const | DM20 | namespace and compile-time linkage |
| 102 | const | DM21 | namespace and compile-time linkage |
| 103 | const | DM25 | namespace and compile-time linkage |
| 104 | const | DM26 | namespace and compile-time linkage |
| 105 | const | DM27 | namespace and compile-time linkage |
| 107 | const | SEC | namespace and compile-time linkage |
| 108 | const | SECACK | namespace and compile-time linkage |
| 114 | struct | SpnDef | data/schema coupling and match/serde break risk |
| 125 | struct | PgnDef | data/schema coupling and match/serde break risk |
| 1112 | fn | find_pgn | API and behavior coupling to callers/implementers |
| 1119 | struct | J1939Frame | data/schema coupling and match/serde break risk |
| 1132 | struct | SpnValue | data/schema coupling and match/serde break risk |
| 1139 | impl | J1939Frame | method behavior and trait semantics |
| 1140 | fn | from_raw | API and behavior coupling to callers/implementers |
| 1172 | fn | build_id | API and behavior coupling to callers/implementers |
| 1179 | fn | decode | API and behavior coupling to callers/implementers |
| 1203 | fn | pgn_name | API and behavior coupling to callers/implementers |
| 1207 | fn | sa_name | API and behavior coupling to callers/implementers |
| 1227 | fn | data_hex | API and behavior coupling to callers/implementers |
| 1236 | fn | bits | API and behavior coupling to callers/implementers |
| 1251 | struct | Builder | data/schema coupling and match/serde break risk |
| 1252 | impl | Builder | method behavior and trait semantics |
| 1253 | fn | eec1 | API and behavior coupling to callers/implementers |
| 1265 | fn | eec2 | API and behavior coupling to callers/implementers |
| 1271 | fn | et1 | API and behavior coupling to callers/implementers |
| 1280 | fn | efl_p1 | API and behavior coupling to callers/implementers |
| 1287 | fn | ic1 | API and behavior coupling to callers/implementers |
| 1296 | fn | lfe | API and behavior coupling to callers/implementers |
| 1304 | fn | pto | API and behavior coupling to callers/implementers |
| 1312 | fn | ccvs | API and behavior coupling to callers/implementers |
| 1322 | fn | dm1 | API and behavior coupling to callers/implementers |
| 1342 | fn | hours | API and behavior coupling to callers/implementers |
| 1353 | fn | adas1 | API and behavior coupling to callers/implementers |
| 1377 | fn | fus1 | API and behavior coupling to callers/implementers |
| 1395 | fn | engy1 | API and behavior coupling to callers/implementers |
| 1423 | struct | Dtc | data/schema coupling and match/serde break risk |
| 1433 | enum | DtcSeverity | data/schema coupling and match/serde break risk |
| 1440 | impl | std::fmt::Display for DtcSeverity | method behavior and trait semantics |
| 1441 | fn | fmt | API and behavior coupling to callers/implementers |
| 1451 | fn | fmi_name | API and behavior coupling to callers/implementers |
| 1480 | struct | J1939Bus | data/schema coupling and match/serde break risk |
| 1491 | impl | Default for J1939Bus | method behavior and trait semantics |
| 1492 | fn | default | API and behavior coupling to callers/implementers |
| 1497 | impl | J1939Bus | method behavior and trait semantics |
| 1498 | fn | new | API and behavior coupling to callers/implementers |
| 1511 | fn | push | API and behavior coupling to callers/implementers |
| 1520 | fn | tick | API and behavior coupling to callers/implementers |
| 1531 | fn | visible_frames | API and behavior coupling to callers/implementers |
| 1540 | module | tests | namespace and compile-time linkage |
| 1544 | fn | pdu1_id_roundtrip_preserves_destination | API and behavior coupling to callers/implementers |
| 1553 | fn | ebc2_decodes_all_wheel_speeds | API and behavior coupling to callers/implementers |
| 1589 | fn | etc2_gear_decode_matches_encoded_value | API and behavior coupling to callers/implementers |
| 1612 | fn | adas_builder_roundtrip_exposes_core_spns | API and behavior coupling to callers/implementers |
| 1635 | fn | energy_builder_roundtrip_exposes_power_values | API and behavior coupling to callers/implementers |

## Line by Line Change Impact

| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! J1939 Protocol Stack — SAE J1939 / ISO 11783 (ISOBUS) | comment/documentation; changing affects understanding |
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! 29-bit CAN ID layout: | comment/documentation; changing affects understanding and intent traceability. |

```
| 4 | //!  [28:26] Priority (0=highest, 7=lowest) | comment/documentation; changing affects understanding and intent tra
| 5 | //!  [25]    Reserved | comment/documentation; changing affects understanding and intent traceability. |
| 6 | //!  [24]    Data Page (DP) | comment/documentation; changing affects understanding and intent traceability. |
| 7 | //!  [23:16] PDU Format (PF) | comment/documentation; changing affects understanding and intent traceability. |
| 8 | //!  [15:8]  PDU Specific (PS) — DA if PF<0xF0, group-ext if PF≥0xF0 | comment/documentation; changing affects und
| 9 | //!  [7:0]   Source Address (SA) | comment/documentation; changing affects understanding and intent traceability.
| 10 | (blank) | blank separator; no runtime behavior. |
| 11 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | // ■■ Source Address Constants ■■■
```

```
| 80 |     pub const TCFG: u32 = 65092; // Transmission Configuration | support logic; impact depends on neighboring blo
| 81 |     pub const EC1: u32 = 61448; // Electronic Clutch Controller 1 | support logic; impact depends on neighboring
| 82 |     pub const HRVD: u32 = 65264; // High Resolution Vehicle Distance | support logic; impact depends on neighbori
| 83 |     pub const SHUTDN: u32 = 65252; // Shutdown | support logic; impact depends on neighboring block and data flow
| 84 |     pub const SOFT: u32 = 65251; // Software Identification | support logic; impact depends on neighboring block
| 85 |     pub const ECFG: u32 = 57600; // Engine Configuration | support logic; impact depends on neighboring block and
| 86 |     pub const PTODE: u32 = 57088; // PTO Drive Engagement | support logic; impact depends on neighboring block an
| 87 |     pub const TD: u32 = 65254; // Time / Date | support logic; impact depends on neighboring block and data flow
| 88 |     pub const TIRE: u32 = 65268; // Tire condition (TPMS) | support logic; impact depends on neighboring block an
| 89 |     pub const EI: u32 = 65255; // Engine Information | support logic; impact depends on neighboring block and dat
| 90 |     pub const VEP1: u32 = 65271; // Vehicle Electrical Power 1 | support logic; impact depends on neighboring blo
| 91 |     pub const SERV: u32 = 65226; // Service (same PGN space — DM1 alias) | support logic; impact depends on neigh
| 92 |     pub const EBC2: u32 = 65215; // Electronic Brake Controller 2 (wheel speeds) | support logic; impact depends
| 93 |                                  // J1939-75: Generator sets | comment/documentation; changing affects understand
| 94 |     pub const GSC1: u32 = 65008; // Generator AC Voltage/Current | support logic; impact depends on neighboring b
| 95 |                                  // Network Management | comment/documentation; changing affects understanding an
| 96 |     pub const NM_ACK: u32 = 59392; // NM Acknowledgement | support logic; impact depends on neighboring block and
| 97 |     pub const DM13: u32 = 49664; // Stop/Start Broadcast (NM coordination) | support logic; impact depends on nei
| 98 |     pub const DM14: u32 = 49920; // Memory Access Request | support logic; impact depends on neighboring block an
| 99 |     pub const DM18: u32 = 54528; // Data Security | support logic; impact depends on neighboring block and data f
| 100 |     pub const DM19: u32 = 54784; // Calibration Information | support logic; impact depends on neighboring bloc
| 101 |     pub const DM20: u32 = 49152; // Monitor Performance Ratio | support logic; impact depends on neighboring blo
| 102 |     pub const DM21: u32 = 49408; // Diagnostic Readiness 2 | support logic; impact depends on neighboring block
| 103 |     pub const DM25: u32 = 57088; // Expanded Freeze Frame | support logic; impact depends on neighboring block a
| 104 |     pub const DM26: u32 = 57344; // Diagnostic Readiness 3 | support logic; impact depends on neighboring block
| 105 |     pub const DM27: u32 = 57600; // All Pending DTCs | support logic; impact depends on neighboring block and da
| 106 |                                  // J1939-76 Security | comment/documentation; changing affects understanding an
| 107 |     pub const SEC: u32 = 57856; // Security Key Exchange | support logic; impact depends on neighboring block an
| 108 |     pub const SECACK: u32 = 58112; // Security Acknowledgement | support logic; impact depends on neighboring bl
| 109 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 110 | (blank) | blank separator; no runtime behavior. |
| 111 | // ■■ SPN/PGN Registry ■■■
```

```
| 156 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 157 |                 offset: -125.0, | support logic; impact depends on neighboring block and data flow. |
| 158 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 159 |             }, | support logic; impact depends on neighboring block and data flow. |
| 160 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 161 |                 spn: 513, | support logic; impact depends on neighboring block and data flow. |
| 162 |                 name: "Actual Engine Torque", | support logic; impact depends on neighboring block and data flo
| 163 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 164 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 165 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 166 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 167 |                 offset: -125.0, | support logic; impact depends on neighboring block and data flow. |
| 168 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 169 |             }, | support logic; impact depends on neighboring block and data flow. |
| 170 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 171 |                 spn: 190, | support logic; impact depends on neighboring block and data flow. |
| 172 |                 name: "Engine Speed", | support logic; impact depends on neighboring block and data flow. |
| 173 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 174 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 175 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 176 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 177 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 178 |                 unit: "rpm", | support logic; impact depends on neighboring block and data flow. |
| 179 |             }, | support logic; impact depends on neighboring block and data flow. |
| 180 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 181 |                 spn: 1675, | support logic; impact depends on neighboring block and data flow. |
| 182 |                 name: "Engine Starter Mode", | support logic; impact depends on neighboring block and data flow
| 183 |                 byte_offset: 6, | support logic; impact depends on neighboring block and data flow. |
| 184 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 185 |                 bit_length: 4, | support logic; impact depends on neighboring block and data flow. |
| 186 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 187 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 188 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 189 |             }, | support logic; impact depends on neighboring block and data flow. |
| 190 |         ], | support logic; impact depends on neighboring block and data flow. |
| 191 |     }, | support logic; impact depends on neighboring block and data flow. |
| 192 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 193 |         pgn: 61445, | support logic; impact depends on neighboring block and data flow. |
| 194 |         name: "EEC2", | support logic; impact depends on neighboring block and data flow. |
| 195 |         desc: "Electronic Engine Control 2", | support logic; impact depends on neighboring block and data flow
| 196 |         rate_ms: 50, | support logic; impact depends on neighboring block and data flow. |
| 197 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 198 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 199 |                 spn: 558, | support logic; impact depends on neighboring block and data flow. |
| 200 |                 name: "Accel Pedal Pos 1", | support logic; impact depends on neighboring block and data flow.
| 201 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 202 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 203 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 204 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 205 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 206 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 207 |             }, | support logic; impact depends on neighboring block and data flow. |
| 208 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 209 |                 spn: 91, | support logic; impact depends on neighboring block and data flow. |
| 210 |                 name: "Throttle Position", | support logic; impact depends on neighboring block and data flow.
| 211 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 212 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 213 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 214 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 215 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 216 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 217 |             }, | support logic; impact depends on neighboring block and data flow. |
| 218 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 219 |                 spn: 92, | support logic; impact depends on neighboring block and data flow. |
| 220 |                 name: "Engine Percent Load", | support logic; impact depends on neighboring block and data flow
| 221 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 222 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 223 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 224 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 225 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 226 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 227 |             }, | support logic; impact depends on neighboring block and data flow. |
| 228 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 229 |                 spn: 974, | support logic; impact depends on neighboring block and data flow. |
| 230 |                 name: "Remote Accel Pedal", | support logic; impact depends on neighboring block and data flow.
| 231 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
```

```
| 232 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 233 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 234 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 235 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 236 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 237 |             }, | support logic; impact depends on neighboring block and data flow. |
| 238 |         ], | support logic; impact depends on neighboring block and data flow. |
| 239 |     }, | support logic; impact depends on neighboring block and data flow. |
| 240 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 241 |         pgn: 61442, | support logic; impact depends on neighboring block and data flow. |
| 242 |         name: "ETC1", | support logic; impact depends on neighboring block and data flow. |
| 243 |         desc: "Electronic Transmission Control 1", | support logic; impact depends on neighboring block and data
| 244 |         rate_ms: 20, | support logic; impact depends on neighboring block and data flow. |
| 245 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 246 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 247 |                 spn: 560, | support logic; impact depends on neighboring block and data flow. |
| 248 |                 name: "Trans Driveline Engaged", | support logic; impact depends on neighboring block and data
| 249 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 250 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 251 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 252 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 253 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 254 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 255 |             }, | support logic; impact depends on neighboring block and data flow. |
| 256 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 257 |                 spn: 573, | support logic; impact depends on neighboring block and data flow. |
| 258 |                 name: "Torque Conv Lockup", | support logic; impact depends on neighboring block and data flow.
| 259 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 260 |                 bit_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 261 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 262 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 263 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 264 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 265 |             }, | support logic; impact depends on neighboring block and data flow. |
| 266 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 267 |                 spn: 191, | support logic; impact depends on neighboring block and data flow. |
| 268 |                 name: "Output Shaft Speed", | support logic; impact depends on neighboring block and data flow.
| 269 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 270 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 271 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 272 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 273 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 274 |                 unit: "rpm", | support logic; impact depends on neighboring block and data flow. |
| 275 |             }, | support logic; impact depends on neighboring block and data flow. |
| 276 |         ], | support logic; impact depends on neighboring block and data flow. |
| 277 |     }, | support logic; impact depends on neighboring block and data flow. |
| 278 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 279 |         pgn: 61443, | support logic; impact depends on neighboring block and data flow. |
| 280 |         name: "ETC2", | support logic; impact depends on neighboring block and data flow. |
| 281 |         desc: "Electronic Transmission Control 2", | support logic; impact depends on neighboring block and data
| 282 |         rate_ms: 20, | support logic; impact depends on neighboring block and data flow. |
| 283 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 284 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 285 |                 spn: 524, | support logic; impact depends on neighboring block and data flow. |
| 286 |                 name: "Selected Gear", | support logic; impact depends on neighboring block and data flow. |
| 287 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 288 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 289 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 290 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 291 |                 offset: -125.0, | support logic; impact depends on neighboring block and data flow. |
| 292 |                 unit: "idx", | support logic; impact depends on neighboring block and data flow. |
| 293 |             }, | support logic; impact depends on neighboring block and data flow. |
| 294 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 295 |                 spn: 523, | support logic; impact depends on neighboring block and data flow. |
| 296 |                 name: "Current Gear", | support logic; impact depends on neighboring block and data flow. |
| 297 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 298 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 299 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 300 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 301 |                 offset: -125.0, | support logic; impact depends on neighboring block and data flow. |
| 302 |                 unit: "idx", | support logic; impact depends on neighboring block and data flow. |
| 303 |             }, | support logic; impact depends on neighboring block and data flow. |
| 304 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 305 |                 spn: 525, | support logic; impact depends on neighboring block and data flow. |
| 306 |                 name: "Current Range", | support logic; impact depends on neighboring block and data flow. |
| 307 |                 byte_offset: 7, | support logic; impact depends on neighboring block and data flow. |
```

```
| 308 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 309 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 310 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 311 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 312 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 313 |             }, | support logic; impact depends on neighboring block and data flow. |
| 314 |         ], | support logic; impact depends on neighboring block and data flow. |
| 315 |     }, | support logic; impact depends on neighboring block and data flow. |
| 316 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 317 |         pgn: 61441, | support logic; impact depends on neighboring block and data flow. |
| 318 |         name: "EBC1", | support logic; impact depends on neighboring block and data flow. |
| 319 |         desc: "Electronic Brake Controller 1", | support logic; impact depends on neighboring block and data flo
| 320 |         rate_ms: 20, | support logic; impact depends on neighboring block and data flow. |
| 321 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 322 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 323 |                 spn: 561, | support logic; impact depends on neighboring block and data flow. |
| 324 |                 name: "ABS Active", | support logic; impact depends on neighboring block and data flow. |
| 325 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 326 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 327 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 328 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 329 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 330 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 331 |             }, | support logic; impact depends on neighboring block and data flow. |
| 332 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 333 |                 spn: 562, | support logic; impact depends on neighboring block and data flow. |
| 334 |                 name: "ABS Offroad Switch", | support logic; impact depends on neighboring block and data flow.
| 335 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 336 |                 bit_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 337 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 338 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 339 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 340 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 341 |             }, | support logic; impact depends on neighboring block and data flow. |
| 342 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 343 |                 spn: 563, | support logic; impact depends on neighboring block and data flow. |
| 344 |                 name: "ASR Brake Control", | support logic; impact depends on neighboring block and data flow.
| 345 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 346 |                 bit_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 347 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 348 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 349 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 350 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 351 |             }, | support logic; impact depends on neighboring block and data flow. |
| 352 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 353 |                 spn: 1121, | support logic; impact depends on neighboring block and data flow. |
| 354 |                 name: "ESP Active", | support logic; impact depends on neighboring block and data flow. |
| 355 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 356 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 357 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 358 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 359 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 360 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 361 |             }, | support logic; impact depends on neighboring block and data flow. |
| 362 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 363 |                 spn: 521, | support logic; impact depends on neighboring block and data flow. |
| 364 |                 name: "Front Axle Brake Demand", | support logic; impact depends on neighboring block and data f
| 365 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 366 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 367 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 368 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 369 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 370 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 371 |             }, | support logic; impact depends on neighboring block and data flow. |
| 372 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 373 |                 spn: 522, | support logic; impact depends on neighboring block and data flow. |
| 374 |                 name: "Rear Axle Brake Demand", | support logic; impact depends on neighboring block and data fl
| 375 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 376 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 377 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 378 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 379 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 380 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 381 |             }, | support logic; impact depends on neighboring block and data flow. |
| 382 |         ], | support logic; impact depends on neighboring block and data flow. |
| 383 |     }, | support logic; impact depends on neighboring block and data flow. |
```

```
| 384 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 385 |         pgn: 65215, | support logic; impact depends on neighboring block and data flow. |
| 386 |         name: "EBC2", | support logic; impact depends on neighboring block and data flow. |
| 387 |         desc: "Electronic Brake Controller 2", | support logic; impact depends on neighboring block and data flo
| 388 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 389 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 390 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 391 |                 spn: 904, | support logic; impact depends on neighboring block and data flow. |
| 392 |                 name: "Wheel Speed FL", | support logic; impact depends on neighboring block and data flow. |
| 393 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 394 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 395 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 396 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 397 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 398 |                 unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 399 |             }, | support logic; impact depends on neighboring block and data flow. |
| 400 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 401 |                 spn: 905, | support logic; impact depends on neighboring block and data flow. |
| 402 |                 name: "Wheel Speed FR", | support logic; impact depends on neighboring block and data flow. |
| 403 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 404 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 405 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 406 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 407 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 408 |                 unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 409 |             }, | support logic; impact depends on neighboring block and data flow. |
| 410 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 411 |                 spn: 906, | support logic; impact depends on neighboring block and data flow. |
| 412 |                 name: "Wheel Speed RL", | support logic; impact depends on neighboring block and data flow. |
| 413 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 414 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 415 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 416 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 417 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 418 |                 unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 419 |             }, | support logic; impact depends on neighboring block and data flow. |
| 420 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 421 |                 spn: 907, | support logic; impact depends on neighboring block and data flow. |
| 422 |                 name: "Wheel Speed RR", | support logic; impact depends on neighboring block and data flow. |
| 423 |                 byte_offset: 6, | support logic; impact depends on neighboring block and data flow. |
| 424 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 425 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 426 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 427 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 428 |                 unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 429 |             }, | support logic; impact depends on neighboring block and data flow. |
| 430 |         ], | support logic; impact depends on neighboring block and data flow. |
| 431 |     }, | support logic; impact depends on neighboring block and data flow. |
| 432 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 433 |         pgn: 65262, | support logic; impact depends on neighboring block and data flow. |
| 434 |         name: "ET1", | support logic; impact depends on neighboring block and data flow. |
| 435 |         desc: "Engine Temperature 1", | support logic; impact depends on neighboring block and data flow. |
| 436 |         rate_ms: 1000, | support logic; impact depends on neighboring block and data flow. |
| 437 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 438 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 439 |                 spn: 110, | support logic; impact depends on neighboring block and data flow. |
| 440 |                 name: "Engine Coolant Temp", | support logic; impact depends on neighboring block and data flow
| 441 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 442 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 443 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 444 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 445 |                 offset: -40.0, | support logic; impact depends on neighboring block and data flow. |
| 446 |                 unit: "°C", | support logic; impact depends on neighboring block and data flow. |
| 447 |             }, | support logic; impact depends on neighboring block and data flow. |
| 448 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 449 |                 spn: 174, | support logic; impact depends on neighboring block and data flow. |
| 450 |                 name: "Fuel Temperature", | support logic; impact depends on neighboring block and data flow. |
| 451 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 452 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 453 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 454 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 455 |                 offset: -40.0, | support logic; impact depends on neighboring block and data flow. |
| 456 |                 unit: "°C", | support logic; impact depends on neighboring block and data flow. |
| 457 |             }, | support logic; impact depends on neighboring block and data flow. |
| 458 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 459 |                 spn: 175, | support logic; impact depends on neighboring block and data flow. |
```

```
| 460 |                 name: "Engine Oil Temperature", | support logic; impact depends on neighboring block and data f
| 461 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 462 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 463 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 464 |                 factor: 0.03125, | support logic; impact depends on neighboring block and data flow. |
| 465 |                 offset: -273.0, | support logic; impact depends on neighboring block and data flow. |
| 466 |                 unit: "°C", | support logic; impact depends on neighboring block and data flow. |
| 467 |             }, | support logic; impact depends on neighboring block and data flow. |
| 468 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 469 |                 spn: 176, | support logic; impact depends on neighboring block and data flow. |
| 470 |                 name: "Turbo Oil Temperature", | support logic; impact depends on neighboring block and data flo
| 471 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 472 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 473 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 474 |                 factor: 0.03125, | support logic; impact depends on neighboring block and data flow. |
| 475 |                 offset: -273.0, | support logic; impact depends on neighboring block and data flow. |
| 476 |                 unit: "°C", | support logic; impact depends on neighboring block and data flow. |
| 477 |             }, | support logic; impact depends on neighboring block and data flow. |
| 478 |         ], | support logic; impact depends on neighboring block and data flow. |
| 479 |     }, | support logic; impact depends on neighboring block and data flow. |
| 480 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 481 |         pgn: 65263, | support logic; impact depends on neighboring block and data flow. |
| 482 |         name: "EFL/P1", | support logic; impact depends on neighboring block and data flow. |
| 483 |         desc: "Engine Fluid Level/Pressure 1", | support logic; impact depends on neighboring block and data flo
| 484 |         rate_ms: 500, | support logic; impact depends on neighboring block and data flow. |
| 485 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 486 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 487 |                 spn: 94, | support logic; impact depends on neighboring block and data flow. |
| 488 |                 name: "Fuel Delivery Pressure", | support logic; impact depends on neighboring block and data f
| 489 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 490 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 491 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 492 |                 factor: 4.0, | support logic; impact depends on neighboring block and data flow. |
| 493 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 494 |                 unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 495 |             }, | support logic; impact depends on neighboring block and data flow. |
| 496 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 497 |                 spn: 98, | support logic; impact depends on neighboring block and data flow. |
| 498 |                 name: "Engine Oil Level", | support logic; impact depends on neighboring block and data flow. |
| 499 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 500 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 501 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 502 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 503 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 504 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 505 |             }, | support logic; impact depends on neighboring block and data flow. |
| 506 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 507 |                 spn: 100, | support logic; impact depends on neighboring block and data flow. |
| 508 |                 name: "Engine Oil Pressure", | support logic; impact depends on neighboring block and data flow
| 509 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 510 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 511 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 512 |                 factor: 4.0, | support logic; impact depends on neighboring block and data flow. |
| 513 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 514 |                 unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 515 |             }, | support logic; impact depends on neighboring block and data flow. |
| 516 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 517 |                 spn: 109, | support logic; impact depends on neighboring block and data flow. |
| 518 |                 name: "Coolant Pressure", | support logic; impact depends on neighboring block and data flow. |
| 519 |                 byte_offset: 6, | support logic; impact depends on neighboring block and data flow. |
| 520 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 521 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 522 |                 factor: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 523 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 524 |                 unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 525 |             }, | support logic; impact depends on neighboring block and data flow. |
| 526 |         ], | support logic; impact depends on neighboring block and data flow. |
| 527 |     }, | support logic; impact depends on neighboring block and data flow. |
| 528 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 529 |         pgn: 65270, | support logic; impact depends on neighboring block and data flow. |
| 530 |         name: "IC1", | support logic; impact depends on neighboring block and data flow. |
| 531 |         desc: "Inlet/Exhaust Conditions 1", | support logic; impact depends on neighboring block and data flow.
| 532 |         rate_ms: 500, | support logic; impact depends on neighboring block and data flow. |
| 533 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 534 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 535 |                 spn: 102, | support logic; impact depends on neighboring block and data flow. |
```

```
| 536 |                 name: "Boost Pressure", | support logic; impact depends on neighboring block and data flow. |
| 537 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 538 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 539 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 540 |                 factor: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 541 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 542 |                 unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 543 |             }, | support logic; impact depends on neighboring block and data flow. |
| 544 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 545 |                 spn: 105, | support logic; impact depends on neighboring block and data flow. |
| 546 |                 name: "Intake Manifold Temp", | support logic; impact depends on neighboring block and data flow
| 547 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 548 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 549 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 550 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 551 |                 offset: -40.0, | support logic; impact depends on neighboring block and data flow. |
| 552 |                 unit: "°C", | support logic; impact depends on neighboring block and data flow. |
| 553 |             }, | support logic; impact depends on neighboring block and data flow. |
| 554 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 555 |                 spn: 106, | support logic; impact depends on neighboring block and data flow. |
| 556 |                 name: "Air Inlet Pressure", | support logic; impact depends on neighboring block and data flow.
| 557 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 558 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 559 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 560 |                 factor: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 561 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 562 |                 unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 563 |             }, | support logic; impact depends on neighboring block and data flow. |
| 564 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 565 |                 spn: 107, | support logic; impact depends on neighboring block and data flow. |
| 566 |                 name: "Air Filter Diff Pressure", | support logic; impact depends on neighboring block and data
| 567 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 568 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 569 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 570 |                 factor: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 571 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 572 |                 unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 573 |             }, | support logic; impact depends on neighboring block and data flow. |
| 574 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 575 |                 spn: 173, | support logic; impact depends on neighboring block and data flow. |
| 576 |                 name: "Exhaust Gas Temperature", | support logic; impact depends on neighboring block and data
| 577 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 578 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 579 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 580 |                 factor: 0.03125, | support logic; impact depends on neighboring block and data flow. |
| 581 |                 offset: -273.0, | support logic; impact depends on neighboring block and data flow. |
| 582 |                 unit: "°C", | support logic; impact depends on neighboring block and data flow. |
| 583 |             }, | support logic; impact depends on neighboring block and data flow. |
| 584 |         ], | support logic; impact depends on neighboring block and data flow. |
| 585 |     }, | support logic; impact depends on neighboring block and data flow. |
| 586 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 587 |         pgn: 65266, | support logic; impact depends on neighboring block and data flow. |
| 588 |         name: "LFE", | support logic; impact depends on neighboring block and data flow. |
| 589 |         desc: "Fuel Economy", | support logic; impact depends on neighboring block and data flow. |
| 590 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 591 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 592 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 593 |                 spn: 183, | support logic; impact depends on neighboring block and data flow. |
| 594 |                 name: "Fuel Rate", | support logic; impact depends on neighboring block and data flow. |
| 595 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 596 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 597 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 598 |                 factor: 0.05, | support logic; impact depends on neighboring block and data flow. |
| 599 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 600 |                 unit: "L/h", | support logic; impact depends on neighboring block and data flow. |
| 601 |             }, | support logic; impact depends on neighboring block and data flow. |
| 602 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 603 |                 spn: 184, | support logic; impact depends on neighboring block and data flow. |
| 604 |                 name: "Instant Fuel Economy", | support logic; impact depends on neighboring block and data flow
| 605 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 606 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 607 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 608 |                 factor: 1.0 / 512.0, | support logic; impact depends on neighboring block and data flow. |
| 609 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 610 |                 unit: "km/L", | support logic; impact depends on neighboring block and data flow. |
| 611 |             }, | support logic; impact depends on neighboring block and data flow. |
```

```
| 612 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 613 |                 spn: 51, | support logic; impact depends on neighboring block and data flow. |
| 614 |                 name: "Throttle Position", | support logic; impact depends on neighboring block and data flow.
| 615 |                 byte_offset: 6, | support logic; impact depends on neighboring block and data flow. |
| 616 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 617 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 618 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 619 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 620 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 621 |             }, | support logic; impact depends on neighboring block and data flow. |
| 622 |         ], | support logic; impact depends on neighboring block and data flow. |
| 623 |     }, | support logic; impact depends on neighboring block and data flow. |
| 624 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 625 |         pgn: 65253, | support logic; impact depends on neighboring block and data flow. |
| 626 |         name: "HOURS", | support logic; impact depends on neighboring block and data flow. |
| 627 |         desc: "Engine Hours/Revolutions", | support logic; impact depends on neighboring block and data flow. |
| 628 |         rate_ms: 1000, | support logic; impact depends on neighboring block and data flow. |
| 629 |         spns: &[SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 630 |             spn: 247, | support logic; impact depends on neighboring block and data flow. |
| 631 |             name: "Total Engine Hours", | support logic; impact depends on neighboring block and data flow. |
| 632 |             byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 633 |             bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 634 |             bit_length: 32, | support logic; impact depends on neighboring block and data flow. |
| 635 |             factor: 0.05, | support logic; impact depends on neighboring block and data flow. |
| 636 |             offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 637 |             unit: "h", | support logic; impact depends on neighboring block and data flow. |
| 638 |         }], | support logic; impact depends on neighboring block and data flow. |
| 639 |     }, | support logic; impact depends on neighboring block and data flow. |
| 640 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 641 |         pgn: 65265, | support logic; impact depends on neighboring block and data flow. |
| 642 |         name: "CCVS", | support logic; impact depends on neighboring block and data flow. |
| 643 |         desc: "Cruise Control/Vehicle Speed", | support logic; impact depends on neighboring block and data flow
| 644 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 645 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 646 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 647 |                 spn: 70, | support logic; impact depends on neighboring block and data flow. |
| 648 |                 name: "Parking Brake Switch", | support logic; impact depends on neighboring block and data flow
| 649 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 650 |                 bit_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 651 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 652 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 653 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 654 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 655 |             }, | support logic; impact depends on neighboring block and data flow. |
| 656 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 657 |                 spn: 84, | support logic; impact depends on neighboring block and data flow. |
| 658 |                 name: "Wheel-Based Speed", | support logic; impact depends on neighboring block and data flow.
| 659 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 660 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 661 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 662 |                 factor: 1.0 / 256.0, | support logic; impact depends on neighboring block and data flow. |
| 663 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 664 |                 unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 665 |             }, | support logic; impact depends on neighboring block and data flow. |
| 666 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 667 |                 spn: 85, | support logic; impact depends on neighboring block and data flow. |
| 668 |                 name: "Cruise Control Active", | support logic; impact depends on neighboring block and data flo
| 669 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 670 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 671 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 672 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 673 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 674 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 675 |             }, | support logic; impact depends on neighboring block and data flow. |
| 676 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 677 |                 spn: 86, | support logic; impact depends on neighboring block and data flow. |
| 678 |                 name: "CC Set Speed", | support logic; impact depends on neighboring block and data flow. |
| 679 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 680 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 681 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 682 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 683 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 684 |                 unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 685 |             }, | support logic; impact depends on neighboring block and data flow. |
| 686 |         ], | support logic; impact depends on neighboring block and data flow. |
| 687 |     }, | support logic; impact depends on neighboring block and data flow. |
```

```
| 688 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 689 |         pgn: 65093, | support logic; impact depends on neighboring block and data flow. |
| 690 |         name: "PTO", | support logic; impact depends on neighboring block and data flow. |
| 691 |         desc: "Power Take-Off Info", | support logic; impact depends on neighboring block and data flow. |
| 692 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 693 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 694 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 695 |                 spn: 1691, | support logic; impact depends on neighboring block and data flow. |
| 696 |                 name: "Rear PTO State", | support logic; impact depends on neighboring block and data flow. |
| 697 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 698 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 699 |                 bit_length: 5, | support logic; impact depends on neighboring block and data flow. |
| 700 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 701 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 702 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 703 |             }, | support logic; impact depends on neighboring block and data flow. |
| 704 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 705 |                 spn: 900, | support logic; impact depends on neighboring block and data flow. |
| 706 |                 name: "PTO Engagement Control", | support logic; impact depends on neighboring block and data f
| 707 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 708 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 709 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 710 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 711 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 712 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 713 |             }, | support logic; impact depends on neighboring block and data flow. |
| 714 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 715 |                 spn: 1693, | support logic; impact depends on neighboring block and data flow. |
| 716 |                 name: "Rear PTO Output RPM", | support logic; impact depends on neighboring block and data flow
| 717 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 718 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 719 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 720 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 721 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 722 |                 unit: "rpm", | support logic; impact depends on neighboring block and data flow. |
| 723 |             }, | support logic; impact depends on neighboring block and data flow. |
| 724 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 725 |                 spn: 1694, | support logic; impact depends on neighboring block and data flow. |
| 726 |                 name: "Front PTO Output RPM", | support logic; impact depends on neighboring block and data flo
| 727 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 728 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 729 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 730 |                 factor: 0.125, | support logic; impact depends on neighboring block and data flow. |
| 731 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 732 |                 unit: "rpm", | support logic; impact depends on neighboring block and data flow. |
| 733 |             }, | support logic; impact depends on neighboring block and data flow. |
| 734 |         ], | support logic; impact depends on neighboring block and data flow. |
| 735 |     }, | support logic; impact depends on neighboring block and data flow. |
| 736 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 737 |         pgn: 65091, | support logic; impact depends on neighboring block and data flow. |
| 738 |         name: "HITCH", | support logic; impact depends on neighboring block and data flow. |
| 739 |         desc: "Hitch / Hydraulic Status", | support logic; impact depends on neighboring block and data flow. |
| 740 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 741 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 742 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 743 |                 spn: 50001, | support logic; impact depends on neighboring block and data flow. |
| 744 |                 name: "Hydraulic System Pressure", | support logic; impact depends on neighboring block and data
| 745 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 746 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 747 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 748 |                 factor: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 749 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 750 |                 unit: "bar", | support logic; impact depends on neighboring block and data flow. |
| 751 |             }, | support logic; impact depends on neighboring block and data flow. |
| 752 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 753 |                 spn: 50002, | support logic; impact depends on neighboring block and data flow. |
| 754 |                 name: "Hydraulic Fluid Temp", | support logic; impact depends on neighboring block and data flo
| 755 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 756 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 757 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 758 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 759 |                 offset: -40.0, | support logic; impact depends on neighboring block and data flow. |
| 760 |                 unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 761 |             }, | support logic; impact depends on neighboring block and data flow. |
| 762 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 763 |                 spn: 50003, | support logic; impact depends on neighboring block and data flow. |
```

```
| 764 |                 name: "Pump Flow", | support logic; impact depends on neighboring block and data flow. |
| 765 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 766 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 767 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 768 |                 factor: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 769 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 770 |                 unit: "L/min", | support logic; impact depends on neighboring block and data flow. |
| 771 |             }, | support logic; impact depends on neighboring block and data flow. |
| 772 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 773 |                 spn: 50004, | support logic; impact depends on neighboring block and data flow. |
| 774 |                 name: "Hydraulic Alarm Flags", | support logic; impact depends on neighboring block and data flo
| 775 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 776 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 777 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 778 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 779 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 780 |                 unit: "bits", | support logic; impact depends on neighboring block and data flow. |
| 781 |             }, | support logic; impact depends on neighboring block and data flow. |
| 782 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 783 |                 spn: 50005, | support logic; impact depends on neighboring block and data flow. |
| 784 |                 name: "Hitch Position", | support logic; impact depends on neighboring block and data flow. |
| 785 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 786 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 787 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 788 |                 factor: 100.0 / 255.0, | support logic; impact depends on neighboring block and data flow. |
| 789 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 790 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 791 |             }, | support logic; impact depends on neighboring block and data flow. |
| 792 |         ], | support logic; impact depends on neighboring block and data flow. |
| 793 |     }, | support logic; impact depends on neighboring block and data flow. |
| 794 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 795 |         pgn: 65269, | support logic; impact depends on neighboring block and data flow. |
| 796 |         name: "AMB", | support logic; impact depends on neighboring block and data flow. |
| 797 |         desc: "Ambient Conditions", | support logic; impact depends on neighboring block and data flow. |
| 798 |         rate_ms: 1000, | support logic; impact depends on neighboring block and data flow. |
| 799 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 800 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 801 |                 spn: 171, | support logic; impact depends on neighboring block and data flow. |
| 802 |                 name: "Ambient Air Temp", | support logic; impact depends on neighboring block and data flow. |
| 803 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 804 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 805 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 806 |                 factor: 0.03125, | support logic; impact depends on neighboring block and data flow. |
| 807 |                 offset: -273.0, | support logic; impact depends on neighboring block and data flow. |
| 808 |                 unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 809 |             }, | support logic; impact depends on neighboring block and data flow. |
| 810 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 811 |                 spn: 108, | support logic; impact depends on neighboring block and data flow. |
| 812 |                 name: "Barometric Pressure", | support logic; impact depends on neighboring block and data flow
| 813 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 814 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 815 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 816 |                 factor: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 817 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 818 |                 unit: "kPa", | support logic; impact depends on neighboring block and data flow. |
| 819 |             }, | support logic; impact depends on neighboring block and data flow. |
| 820 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 821 |                 spn: 174, | support logic; impact depends on neighboring block and data flow. |
| 822 |                 name: "Cabin / Inlet Temp", | support logic; impact depends on neighboring block and data flow.
| 823 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 824 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 825 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 826 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 827 |                 offset: -40.0, | support logic; impact depends on neighboring block and data flow. |
| 828 |                 unit: "C", | support logic; impact depends on neighboring block and data flow. |
| 829 |             }, | support logic; impact depends on neighboring block and data flow. |
| 830 |         ], | support logic; impact depends on neighboring block and data flow. |
| 831 |     }, | support logic; impact depends on neighboring block and data flow. |
| 832 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 833 |         pgn: 65271, | support logic; impact depends on neighboring block and data flow. |
| 834 |         name: "VEP1", | support logic; impact depends on neighboring block and data flow. |
| 835 |         desc: "Vehicle Electrical Power 1", | support logic; impact depends on neighboring block and data flow.
| 836 |         rate_ms: 500, | support logic; impact depends on neighboring block and data flow. |
| 837 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 838 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 839 |                 spn: 168, | support logic; impact depends on neighboring block and data flow. |
```

```
| 840 |                 name: "Battery Potential", | support logic; impact depends on neighboring block and data flow.
| 841 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 842 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 843 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 844 |                 factor: 0.05, | support logic; impact depends on neighboring block and data flow. |
| 845 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 846 |                 unit: "V", | support logic; impact depends on neighboring block and data flow. |
| 847 |             }, | support logic; impact depends on neighboring block and data flow. |
| 848 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 849 |                 spn: 158, | support logic; impact depends on neighboring block and data flow. |
| 850 |                 name: "Alternator Current", | support logic; impact depends on neighboring block and data flow.
| 851 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 852 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 853 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 854 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 855 |                 offset: -32000.0, | support logic; impact depends on neighboring block and data flow. |
| 856 |                 unit: "A", | support logic; impact depends on neighboring block and data flow. |
| 857 |             }, | support logic; impact depends on neighboring block and data flow. |
| 858 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 859 |                 spn: 167, | support logic; impact depends on neighboring block and data flow. |
| 860 |                 name: "Charging Voltage", | support logic; impact depends on neighboring block and data flow. |
| 861 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 862 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 863 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 864 |                 factor: 0.05, | support logic; impact depends on neighboring block and data flow. |
| 865 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 866 |                 unit: "V", | support logic; impact depends on neighboring block and data flow. |
| 867 |             }, | support logic; impact depends on neighboring block and data flow. |
| 868 |         ], | support logic; impact depends on neighboring block and data flow. |
| 869 |     }, | support logic; impact depends on neighboring block and data flow. |
| 870 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 871 |         pgn: 65280, | support logic; impact depends on neighboring block and data flow. |
| 872 |         name: "ADAS1", | support logic; impact depends on neighboring block and data flow. |
| 873 |         desc: "ADAS status and interventions", | support logic; impact depends on neighboring block and data flo
| 874 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 875 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 876 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 877 |                 spn: 52001, | support logic; impact depends on neighboring block and data flow. |
| 878 |                 name: "LKA Active", | support logic; impact depends on neighboring block and data flow. |
| 879 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 880 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 881 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 882 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 883 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 884 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 885 |             }, | support logic; impact depends on neighboring block and data flow. |
| 886 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 887 |                 spn: 52002, | support logic; impact depends on neighboring block and data flow. |
| 888 |                 name: "ACC Active", | support logic; impact depends on neighboring block and data flow. |
| 889 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 890 |                 bit_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 891 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 892 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 893 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 894 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 895 |             }, | support logic; impact depends on neighboring block and data flow. |
| 896 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 897 |                 spn: 52003, | support logic; impact depends on neighboring block and data flow. |
| 898 |                 name: "AEB Active", | support logic; impact depends on neighboring block and data flow. |
| 899 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 900 |                 bit_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 901 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 902 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 903 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 904 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 905 |             }, | support logic; impact depends on neighboring block and data flow. |
| 906 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 907 |                 spn: 52004, | support logic; impact depends on neighboring block and data flow. |
| 908 |                 name: "Lead Distance", | support logic; impact depends on neighboring block and data flow. |
| 909 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 910 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 911 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 912 |                 factor: 0.1, | support logic; impact depends on neighboring block and data flow. |
| 913 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 914 |                 unit: "m", | support logic; impact depends on neighboring block and data flow. |
| 915 |             }, | support logic; impact depends on neighboring block and data flow. |
```

```
| 916 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 917 |                 spn: 52005, | support logic; impact depends on neighboring block and data flow. |
| 918 |                 name: "TTC", | support logic; impact depends on neighboring block and data flow. |
| 919 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 920 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 921 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 922 |                 factor: 0.01, | support logic; impact depends on neighboring block and data flow. |
| 923 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 924 |                 unit: "s", | support logic; impact depends on neighboring block and data flow. |
| 925 |             }, | support logic; impact depends on neighboring block and data flow. |
| 926 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 927 |                 spn: 52006, | support logic; impact depends on neighboring block and data flow. |
| 928 |                 name: "Speed Limit", | support logic; impact depends on neighboring block and data flow. |
| 929 |                 byte_offset: 5, | support logic; impact depends on neighboring block and data flow. |
| 930 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 931 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 932 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 933 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 934 |                 unit: "km/h", | support logic; impact depends on neighboring block and data flow. |
| 935 |             }, | support logic; impact depends on neighboring block and data flow. |
| 936 |         ], | support logic; impact depends on neighboring block and data flow. |
| 937 |     }, | support logic; impact depends on neighboring block and data flow. |
| 938 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 939 |         pgn: 65281, | support logic; impact depends on neighboring block and data flow. |
| 940 |         name: "FUS1", | support logic; impact depends on neighboring block and data flow. |
| 941 |         desc: "Sensor fusion hazard status", | support logic; impact depends on neighboring block and data flow
| 942 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 943 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 944 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 945 |                 spn: 52101, | support logic; impact depends on neighboring block and data flow. |
| 946 |                 name: "Fused Objects", | support logic; impact depends on neighboring block and data flow. |
| 947 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 948 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 949 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 950 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 951 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 952 |                 unit: "count", | support logic; impact depends on neighboring block and data flow. |
| 953 |             }, | support logic; impact depends on neighboring block and data flow. |
| 954 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 955 |                 spn: 52102, | support logic; impact depends on neighboring block and data flow. |
| 956 |                 name: "Critical TTC", | support logic; impact depends on neighboring block and data flow. |
| 957 |                 byte_offset: 1, | support logic; impact depends on neighboring block and data flow. |
| 958 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 959 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 960 |                 factor: 0.01, | support logic; impact depends on neighboring block and data flow. |
| 961 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 962 |                 unit: "s", | support logic; impact depends on neighboring block and data flow. |
| 963 |             }, | support logic; impact depends on neighboring block and data flow. |
| 964 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 965 |                 spn: 52103, | support logic; impact depends on neighboring block and data flow. |
| 966 |                 name: "Fusion Confidence", | support logic; impact depends on neighboring block and data flow.
| 967 |                 byte_offset: 3, | support logic; impact depends on neighboring block and data flow. |
| 968 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 969 |                 bit_length: 8, | support logic; impact depends on neighboring block and data flow. |
| 970 |                 factor: 0.4, | support logic; impact depends on neighboring block and data flow. |
| 971 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 972 |                 unit: "%", | support logic; impact depends on neighboring block and data flow. |
| 973 |             }, | support logic; impact depends on neighboring block and data flow. |
| 974 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 975 |                 spn: 52104, | support logic; impact depends on neighboring block and data flow. |
| 976 |                 name: "In-Path Hazard", | support logic; impact depends on neighboring block and data flow. |
| 977 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 978 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 979 |                 bit_length: 1, | support logic; impact depends on neighboring block and data flow. |
| 980 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 981 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 982 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 983 |             }, | support logic; impact depends on neighboring block and data flow. |
| 984 |         ], | support logic; impact depends on neighboring block and data flow. |
| 985 |     }, | support logic; impact depends on neighboring block and data flow. |
| 986 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 987 |         pgn: 65282, | support logic; impact depends on neighboring block and data flow. |
| 988 |         name: "ENGY1", | support logic; impact depends on neighboring block and data flow. |
| 989 |         desc: "Vehicle energy balance", | support logic; impact depends on neighboring block and data flow. |
| 990 |         rate_ms: 100, | support logic; impact depends on neighboring block and data flow. |
| 991 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
```

```
| 992 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 993 |                 spn: 52201, | support logic; impact depends on neighboring block and data flow. |
| 994 |                 name: "Battery Voltage", | support logic; impact depends on neighboring block and data flow. |
| 995 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 996 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 997 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 998 |                 factor: 0.05, | support logic; impact depends on neighboring block and data flow. |
| 999 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1000 |                 unit: "V", | support logic; impact depends on neighboring block and data flow. |
| 1001 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1002 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1003 |                 spn: 52202, | support logic; impact depends on neighboring block and data flow. |
| 1004 |                 name: "Alternator Current", | support logic; impact depends on neighboring block and data flow
| 1005 |                 byte_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 1006 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1007 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 1008 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 1009 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1010 |                 unit: "A", | support logic; impact depends on neighboring block and data flow. |
| 1011 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1012 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1013 |                 spn: 52203, | support logic; impact depends on neighboring block and data flow. |
| 1014 |                 name: "Total Electrical Load", | support logic; impact depends on neighboring block and data f
| 1015 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 1016 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1017 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 1018 |                 factor: 0.01, | support logic; impact depends on neighboring block and data flow. |
| 1019 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1020 |                 unit: "kW", | support logic; impact depends on neighboring block and data flow. |
| 1021 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1022 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1023 |                 spn: 52204, | support logic; impact depends on neighboring block and data flow. |
| 1024 |                 name: "Hydraulic Power", | support logic; impact depends on neighboring block and data flow. |
| 1025 |                 byte_offset: 6, | support logic; impact depends on neighboring block and data flow. |
| 1026 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1027 |                 bit_length: 16, | support logic; impact depends on neighboring block and data flow. |
| 1028 |                 factor: 0.01, | support logic; impact depends on neighboring block and data flow. |
| 1029 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1030 |                 unit: "kW", | support logic; impact depends on neighboring block and data flow. |
| 1031 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1032 |         ], | support logic; impact depends on neighboring block and data flow. |
| 1033 |     }, | support logic; impact depends on neighboring block and data flow. |
| 1034 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 1035 |         pgn: 65226, | support logic; impact depends on neighboring block and data flow. |
| 1036 |         name: "DM1", | support logic; impact depends on neighboring block and data flow. |
| 1037 |         desc: "Active Diagnostic Trouble Codes", | support logic; impact depends on neighboring block and data
| 1038 |         rate_ms: 1000, | support logic; impact depends on neighboring block and data flow. |
| 1039 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 1040 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1041 |                 spn: 1213, | support logic; impact depends on neighboring block and data flow. |
| 1042 |                 name: "MIL Lamp", | support logic; impact depends on neighboring block and data flow. |
| 1043 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1044 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1045 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 1046 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 1047 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1048 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 1049 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1050 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1051 |                 spn: 1214, | support logic; impact depends on neighboring block and data flow. |
| 1052 |                 name: "Red Stop Lamp", | support logic; impact depends on neighboring block and data flow. |
| 1053 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1054 |                 bit_offset: 2, | support logic; impact depends on neighboring block and data flow. |
| 1055 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 1056 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 1057 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1058 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 1059 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1060 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1061 |                 spn: 1215, | support logic; impact depends on neighboring block and data flow. |
| 1062 |                 name: "Amber Warning Lamp", | support logic; impact depends on neighboring block and data flow
| 1063 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1064 |                 bit_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 1065 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 1066 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 1067 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 1068 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 1069 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1070 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1071 |                 spn: 1216, | support logic; impact depends on neighboring block and data flow. |
| 1072 |                 name: "Protect Lamp", | support logic; impact depends on neighboring block and data flow. |
| 1073 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1074 |                 bit_offset: 6, | support logic; impact depends on neighboring block and data flow. |
| 1075 |                 bit_length: 2, | support logic; impact depends on neighboring block and data flow. |
| 1076 |                 factor: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 1077 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1078 |                 unit: "", | support logic; impact depends on neighboring block and data flow. |
| 1079 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1080 |         ], | support logic; impact depends on neighboring block and data flow. |
| 1081 |     }, | support logic; impact depends on neighboring block and data flow. |
| 1082 |     PgnDef { | support logic; impact depends on neighboring block and data flow. |
| 1083 |         pgn: 65257, | support logic; impact depends on neighboring block and data flow. |
| 1084 |         name: "FUEL1", | support logic; impact depends on neighboring block and data flow. |
| 1085 |         desc: "Fuel Consumption (cumulative)", | support logic; impact depends on neighboring block and data f
| 1086 |         rate_ms: 1000, | support logic; impact depends on neighboring block and data flow. |
| 1087 |         spns: &[ | support logic; impact depends on neighboring block and data flow. |
| 1088 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1089 |                 spn: 182, | support logic; impact depends on neighboring block and data flow. |
| 1090 |                 name: "Trip Fuel", | support logic; impact depends on neighboring block and data flow. |
| 1091 |                 byte_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1092 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1093 |                 bit_length: 32, | support logic; impact depends on neighboring block and data flow. |
| 1094 |                 factor: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 1095 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1096 |                 unit: "L", | support logic; impact depends on neighboring block and data flow. |
| 1097 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1098 |             SpnDef { | support logic; impact depends on neighboring block and data flow. |
| 1099 |                 spn: 250, | support logic; impact depends on neighboring block and data flow. |
| 1100 |                 name: "Total Fuel Used", | support logic; impact depends on neighboring block and data flow. |
| 1101 |                 byte_offset: 4, | support logic; impact depends on neighboring block and data flow. |
| 1102 |                 bit_offset: 0, | support logic; impact depends on neighboring block and data flow. |
| 1103 |                 bit_length: 32, | support logic; impact depends on neighboring block and data flow. |
| 1104 |                 factor: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 1105 |                 offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1106 |                 unit: "L", | support logic; impact depends on neighboring block and data flow. |
| 1107 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1108 |         ], | support logic; impact depends on neighboring block and data flow. |
| 1109 |     }, | support logic; impact depends on neighboring block and data flow. |
| 1110 | ]; | support logic; impact depends on neighboring block and data flow. |
| 1111 | (blank) | blank separator; no runtime behavior. |
| 1112 | pub fn find_pgn(pgn: u32) -> Option<&'static PgnDef> { | function signature/body; changes affect API behavior a
| 1113 |     REGISTRY.iter().find(\|p\| p.pgn == pgn) | support logic; impact depends on neighboring block and data flow
| 1114 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1115 | (blank) | blank separator; no runtime behavior. |
| 1116 | // ■■ J1939 Frame ■■■
```

```
| 1144 |         let ps = ((raw_id >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data fl
| 1145 |         let sa = (raw_id & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1146 | (blank) | blank separator; no runtime behavior. |
| 1147 |         let (pgn, da) = if pf < 0xF0 { | support logic; impact depends on neighboring block and data flow. |
| 1148 |             (((dp << 17) \| ((pf as u32) << 8)), ps) | support logic; impact depends on neighboring block and d
| 1149 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 1150 |             (((dp << 17) \| ((pf as u32) << 8) \| (ps as u32)), 0xFF) | support logic; impact depends on neighb
| 1151 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1152 | (blank) | blank separator; no runtime behavior. |
| 1153 |         let mut arr = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1154 |         let len = data.len().min(8); | support logic; impact depends on neighboring block and data flow. |
| 1155 |         arr[..len].copy_from_slice(&data[..len]); | support logic; impact depends on neighboring block and data
| 1156 | (blank) | blank separator; no runtime behavior. |
| 1157 |         let mut f = J1939Frame { | support logic; impact depends on neighboring block and data flow. |
| 1158 |             timestamp: ts, | support logic; impact depends on neighboring block and data flow. |
| 1159 |             raw_id, | support logic; impact depends on neighboring block and data flow. |
| 1160 |             priority, | support logic; impact depends on neighboring block and data flow. |
| 1161 |             pgn, | support logic; impact depends on neighboring block and data flow. |
| 1162 |             sa, | support logic; impact depends on neighboring block and data flow. |
| 1163 |             da, | support logic; impact depends on neighboring block and data flow. |
| 1164 |             data: arr, | support logic; impact depends on neighboring block and data flow. |
| 1165 |             dlc: len as u8, | support logic; impact depends on neighboring block and data flow. |
| 1166 |             decoded: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 1167 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1168 |         f.decode(); | support logic; impact depends on neighboring block and data flow. |
| 1169 |         f | support logic; impact depends on neighboring block and data flow. |
| 1170 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1171 | (blank) | blank separator; no runtime behavior. |
| 1172 |     pub fn build_id(priority: u8, pgn: u32, sa: u8, da: u8) -> u32 { | function signature/body; changes affect
| 1173 |         let dp = (pgn >> 17) & 0x1; | support logic; impact depends on neighboring block and data flow. |
| 1174 |         let pf = (pgn >> 8) & 0xFF; | support logic; impact depends on neighboring block and data flow. |
| 1175 |         let ps = if pf < 0xF0 { da as u32 } else { pgn & 0xFF }; | support logic; impact depends on neighboring
| 1176 |         ((priority as u32 & 7) << 26) \| (dp << 24) \| (pf << 16) \| (ps << 8) \| (sa as u32) | support logic;
| 1177 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1178 | (blank) | blank separator; no runtime behavior. |
| 1179 |     fn decode(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1180 |         let def = match find_pgn(self.pgn) { | support logic; impact depends on neighboring block and data flo
| 1181 |             Some(d) => d, | support logic; impact depends on neighboring block and data flow. |
| 1182 |             None => return, | support logic; impact depends on neighboring block and data flow. |
| 1183 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1184 |         for spn in def.spns { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1185 |             let end_byte = | support logic; impact depends on neighboring block and data flow. |
| 1186 |                 spn.byte_offset + (spn.bit_offset as usize + spn.bit_length as usize).div_ceil(8); | support l
| 1187 |             if end_byte > self.dlc as usize { | runtime gate; condition changes can enable/disable critical pa
| 1188 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 1189 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1190 |             let raw = bits(&self.data, spn.byte_offset, spn.bit_offset, spn.bit_length); | support logic; impa
| 1191 |             if raw == (1u64 << spn.bit_length) - 1 { | runtime gate; condition changes can enable/disable crit
| 1192 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 1193 |             } // all-ones = not available | support logic; impact depends on neighboring block and data flow.
| 1194 |             self.decoded.push(SpnValue { | support logic; impact depends on neighboring block and data flow. |
| 1195 |                 spn: spn.spn, | support logic; impact depends on neighboring block and data flow. |
| 1196 |                 name: spn.name, | support logic; impact depends on neighboring block and data flow. |
| 1197 |                 physical: raw as f64 * spn.factor + spn.offset, | support logic; impact depends on neighboring
| 1198 |                 unit: spn.unit, | support logic; impact depends on neighboring block and data flow. |
| 1199 |             }); | support logic; impact depends on neighboring block and data flow. |
| 1200 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1201 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1202 | (blank) | blank separator; no runtime behavior. |
| 1203 |     pub fn pgn_name(&self) -> &'static str { | function signature/body; changes affect API behavior and contro
| 1204 |         find_pgn(self.pgn).map(\|p\| p.name).unwrap_or("???") | error propagation point; changes alter failure
| 1205 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1206 | (blank) | blank separator; no runtime behavior. |
| 1207 |     pub fn sa_name(&self) -> &'static str { | function signature/body; changes affect API behavior and control
| 1208 |         match self.sa { | branch dispatcher; arm changes alter behavior class handling. |
| 1209 |             0x00 => "ECM1 ", | support logic; impact depends on neighboring block and data flow. |
| 1210 |             0x01 => "ECM2 ", | support logic; impact depends on neighboring block and data flow. |
| 1211 |             0x02 => "TURBO", | support logic; impact depends on neighboring block and data flow. |
| 1212 |             0x03 => "TRANS", | support logic; impact depends on neighboring block and data flow. |
| 1213 |             0x0B => "ABS  ", | support logic; impact depends on neighboring block and data flow. |
| 1214 |             0x0D => "TC   ", | support logic; impact depends on neighboring block and data flow. |
| 1215 |             0x1C => "DASH ", | support logic; impact depends on neighboring block and data flow. |
| 1216 |             0x25 => "IMPL ", | support logic; impact depends on neighboring block and data flow. |
| 1217 |             0x26 => "VT   ", | support logic; impact depends on neighboring block and data flow. |
| 1218 |             0x27 => "CAB  ", | support logic; impact depends on neighboring block and data flow. |
| 1219 |             0x28 => "ARST ", | support logic; impact depends on neighboring block and data flow. |
```

```
| 1220 |             0x1E => "HITCH", | support logic; impact depends on neighboring block and data flow. |
| 1221 |             0xFE => "NULL ", | support logic; impact depends on neighboring block and data flow. |
| 1222 |             0xFF => "BCAST", | support logic; impact depends on neighboring block and data flow. |
| 1223 |             _ => "UNK  ", | support logic; impact depends on neighboring block and data flow. |
| 1224 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1225 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1226 | (blank) | blank separator; no runtime behavior. |
| 1227 |     pub fn data_hex(&self) -> String { | function signature/body; changes affect API behavior and control-flow
| 1228 |         self.data[..self.dlc as usize] | support logic; impact depends on neighboring block and data flow. |
| 1229 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1230 |             .map(\|b\| format!("{:02X}", b)) | support logic; impact depends on neighboring block and data flow
| 1231 |             .collect::<Vec<_>>() | support logic; impact depends on neighboring block and data flow. |
| 1232 |             .join(" ") | support logic; impact depends on neighboring block and data flow. |
| 1233 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1234 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1235 | (blank) | blank separator; no runtime behavior. |
| 1236 | fn bits(data: &[u8], byte_off: usize, bit_off: u8, len: u8) -> u64 { | function signature/body; changes affect
| 1237 |     let mut v = 0u64; | support logic; impact depends on neighboring block and data flow. |
| 1238 |     for i in 0..len as usize { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1239 |         let abs = byte_off * 8 + bit_off as usize + i; | support logic; impact depends on neighboring block and
| 1240 |         let b = abs / 8; | support logic; impact depends on neighboring block and data flow. |
| 1241 |         let k = abs % 8; | support logic; impact depends on neighboring block and data flow. |
| 1242 |         if b < data.len() && (data[b] >> k) & 1 == 1 { | runtime gate; condition changes can enable/disable cri
| 1243 |             v \|= 1 << i; | support logic; impact depends on neighboring block and data flow. |
| 1244 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1245 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1246 |     v | support logic; impact depends on neighboring block and data flow. |
| 1247 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1248 | (blank) | blank separator; no runtime behavior. |
| 1249 | // ■■ J1939 Frame Builders ■■■
```

```
| 1296 |     pub fn lfe(ts: f64, fuel_lph: f64, throttle: f64, sa: u8) -> J1939Frame { | function signature/body; change
| 1297 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1298 |         let f = (fuel_lph / 0.05) as u16; | support logic; impact depends on neighboring block and data flow.
| 1299 |         d[0] = (f & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1300 |         d[1] = (f >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1301 |         d[6] = (throttle / 0.4).clamp(0.0, 250.0) as u8; | support logic; impact depends on neighboring block a
| 1302 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::LFE, sa, 0xFF), &d) | support logic; impact deper
| 1303 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1304 |     pub fn pto(ts: f64, rpm: f64, enabled: bool, sa: u8) -> J1939Frame { | function signature/body; changes af
| 1305 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1306 |         d[0] = if enabled { 0x01 } else { 0x00 }; | support logic; impact depends on neighboring block and data
| 1307 |         let r = (rpm / 0.125) as u16; | support logic; impact depends on neighboring block and data flow. |
| 1308 |         d[2] = (r & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1309 |         d[3] = (r >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1310 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::PTO, sa, 0xFF), &d) | support logic; impact deper
| 1311 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1312 |     pub fn ccvs(ts: f64, speed: f64, cc_active: bool, cc_speed: f64, sa: u8) -> J1939Frame { | function signatu
| 1313 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1314 |         d[0] = 0b11111100; | support logic; impact depends on neighboring block and data flow. |
| 1315 |         let s = (speed * 256.0) as u16; | support logic; impact depends on neighboring block and data flow. |
| 1316 |         d[1] = (s & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1317 |         d[2] = (s >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1318 |         d[3] = if cc_active { 0x01 } else { 0x00 }; | support logic; impact depends on neighboring block and da
| 1319 |         d[4] = cc_speed.clamp(0.0, 250.0) as u8; | support logic; impact depends on neighboring block and data
| 1320 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::CCVS, sa, 0xFF), &d) | support logic; impact depe
| 1321 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1322 |     pub fn dm1( | function signature/body; changes affect API behavior and control-flow. |
| 1323 |         ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 1324 |         amber: bool, | support logic; impact depends on neighboring block and data flow. |
| 1325 |         red: bool, | support logic; impact depends on neighboring block and data flow. |
| 1326 |         mil: bool, | support logic; impact depends on neighboring block and data flow. |
| 1327 |         dtc_spn: u32, | support logic; impact depends on neighboring block and data flow. |
| 1328 |         fmi: u8, | support logic; impact depends on neighboring block and data flow. |
| 1329 |         sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 1330 |     ) -> J1939Frame { | support logic; impact depends on neighboring block and data flow. |
| 1331 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1332 |         d[0] = (if mil { 0x40 } else { 0 }) | support logic; impact depends on neighboring block and data flow
| 1333 |             \| (if red { 0x10 } else { 0 }) | support logic; impact depends on neighboring block and data flow
| 1334 |             \| (if amber { 0x04 } else { 0 }); | support logic; impact depends on neighboring block and data f
| 1335 |         d[1] = 0xFF; | support logic; impact depends on neighboring block and data flow. |
| 1336 |         d[2] = (dtc_spn & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1337 |         d[3] = ((dtc_spn >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flo
| 1338 |         d[4] = (((dtc_spn >> 16) & 0x7) as u8) \| ((fmi & 0x1F) << 3); | support logic; impact depends on neigh
| 1339 |         d[5] = 1; // occurrence count | support logic; impact depends on neighboring block and data flow. |
| 1340 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::DM1, sa, 0xFF), &d) | support logic; impact deper
| 1341 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1342 |     pub fn hours(ts: f64, total_h: f64, sa: u8) -> J1939Frame { | function signature/body; changes affect API b
| 1343 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1344 |         let h = (total_h / 0.05) as u32; | support logic; impact depends on neighboring block and data flow. |
| 1345 |         d[0] = (h & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1346 |         d[1] = ((h >> 8) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1347 |         d[2] = ((h >> 16) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1348 |         d[3] = ((h >> 24) & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1349 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::HOURS, sa, 0xFF), &d) | support logic; impact de
| 1350 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1351 | (blank) | blank separator; no runtime behavior. |
| 1352 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent tra
| 1353 |     pub fn adas1( | function signature/body; changes affect API behavior and control-flow. |
| 1354 |         ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 1355 |         lka_on: bool, | support logic; impact depends on neighboring block and data flow. |
| 1356 |         acc_on: bool, | support logic; impact depends on neighboring block and data flow. |
| 1357 |         aeb_on: bool, | support logic; impact depends on neighboring block and data flow. |
| 1358 |         lead_distance_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 1359 |         ttc_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 1360 |         speed_limit_kmh: f64, | support logic; impact depends on neighboring block and data flow. |
| 1361 |         sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 1362 |     ) -> J1939Frame { | support logic; impact depends on neighboring block and data flow. |
| 1363 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1364 |         d[0] = (if lka_on { 0x01 } else { 0x00 }) | support logic; impact depends on neighboring block and data
| 1365 |             \| (if acc_on { 0x01 } else { 0x00 }) << 2 | support logic; impact depends on neighboring block and
| 1366 |             \| (if aeb_on { 0x01 } else { 0x00 }) << 4; | support logic; impact depends on neighboring block a
| 1367 |         let lead = (lead_distance_m.clamp(0.0, 6553.0) / 0.1) as u16; | support logic; impact depends on neigh
| 1368 |         d[1] = (lead & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1369 |         d[2] = (lead >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1370 |         let ttc = (ttc_s.clamp(0.0, 655.35) / 0.01) as u16; | support logic; impact depends on neighboring bloc
| 1371 |         d[3] = (ttc & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
```

```
| 1372 |         d[4] = (ttc >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1373 |         d[5] = speed_limit_kmh.clamp(0.0, 250.0) as u8; | support logic; impact depends on neighboring block an
| 1374 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::ADAS1, sa, 0xFF), &d) | support logic; impact dep
| 1375 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1376 | (blank) | blank separator; no runtime behavior. |
| 1377 |     pub fn fus1( | function signature/body; changes affect API behavior and control-flow. |
| 1378 |         ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 1379 |         fused_objects: usize, | support logic; impact depends on neighboring block and data flow. |
| 1380 |         critical_ttc_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 1381 |         confidence_01: f64, | support logic; impact depends on neighboring block and data flow. |
| 1382 |         in_path_hazard: bool, | support logic; impact depends on neighboring block and data flow. |
| 1383 |         sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 1384 |     ) -> J1939Frame { | support logic; impact depends on neighboring block and data flow. |
| 1385 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1386 |         d[0] = fused_objects.min(255) as u8; | support logic; impact depends on neighboring block and data flo
| 1387 |         let ttc = (critical_ttc_s.clamp(0.0, 655.35) / 0.01) as u16; | support logic; impact depends on neighbo
| 1388 |         d[1] = (ttc & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1389 |         d[2] = (ttc >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1390 |         d[3] = (confidence_01.clamp(0.0, 1.0) * 250.0) as u8; | support logic; impact depends on neighboring b
| 1391 |         d[4] = if in_path_hazard { 0x01 } else { 0x00 }; | support logic; impact depends on neighboring block a
| 1392 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::FUS1, sa, 0xFF), &d) | support logic; impact dep
| 1393 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1394 | (blank) | blank separator; no runtime behavior. |
| 1395 |     pub fn engy1( | function signature/body; changes affect API behavior and control-flow. |
| 1396 |         ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 1397 |         battery_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 1398 |         alternator_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 1399 |         electrical_kw: f64, | support logic; impact depends on neighboring block and data flow. |
| 1400 |         hydraulic_kw: f64, | support logic; impact depends on neighboring block and data flow. |
| 1401 |         sa: u8, | support logic; impact depends on neighboring block and data flow. |
| 1402 |     ) -> J1939Frame { | support logic; impact depends on neighboring block and data flow. |
| 1403 |         let mut d = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1404 |         let bv = (battery_v.clamp(0.0, 3276.75) / 0.05) as u16; | support logic; impact depends on neighboring
| 1405 |         d[0] = (bv & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1406 |         d[1] = (bv >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1407 |         let ia = alternator_a.clamp(0.0, 65535.0) as u16; | support logic; impact depends on neighboring block
| 1408 |         d[2] = (ia & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1409 |         d[3] = (ia >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1410 |         let ek = (electrical_kw.clamp(0.0, 655.35) / 0.01) as u16; | support logic; impact depends on neighbori
| 1411 |         d[4] = (ek & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1412 |         d[5] = (ek >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1413 |         let hk = (hydraulic_kw.clamp(0.0, 655.35) / 0.01) as u16; | support logic; impact depends on neighborin
| 1414 |         d[6] = (hk & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1415 |         d[7] = (hk >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1416 |         J1939Frame::from_raw(ts, J1939Frame::build_id(6, pgn::ENGY1, sa, 0xFF), &d) | support logic; impact dep
| 1417 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1418 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1419 | (blank) | blank separator; no runtime behavior. |
| 1420 | // ■■ DTC Definitions ■■■
```

```
| 1448 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1449 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1450 | (blank) | blank separator; no runtime behavior. |
| 1451 | pub fn fmi_name(fmi: u8) -> &'static str { | function signature/body; changes affect API behavior and control-f
| 1452 |     match fmi { | branch dispatcher; arm changes alter behavior class handling. |
| 1453 |         0 => "Above Normal — Most Severe", | support logic; impact depends on neighboring block and data flow.
| 1454 |         1 => "Below Normal — Most Severe", | support logic; impact depends on neighboring block and data flow.
| 1455 |         2 => "Erratic / Intermittent", | support logic; impact depends on neighboring block and data flow. |
| 1456 |         3 => "Voltage High / Short to High", | support logic; impact depends on neighboring block and data flow
| 1457 |         4 => "Voltage Low / Short to Low", | support logic; impact depends on neighboring block and data flow.
| 1458 |         5 => "Current Low / Open Circuit", | support logic; impact depends on neighboring block and data flow.
| 1459 |         6 => "Current High / Grounded", | support logic; impact depends on neighboring block and data flow. |
| 1460 |         7 => "Mechanical System Not Responding", | support logic; impact depends on neighboring block and data
| 1461 |         8 => "Abnormal Frequency / Pulse Width", | support logic; impact depends on neighboring block and data
| 1462 |         9 => "Abnormal Update Rate", | support logic; impact depends on neighboring block and data flow. |
| 1463 |         10 => "Abnormal Rate of Change", | support logic; impact depends on neighboring block and data flow. |
| 1464 |         11 => "Root Cause Not Known", | support logic; impact depends on neighboring block and data flow. |
| 1465 |         12 => "Bad Device / Component", | support logic; impact depends on neighboring block and data flow. |
| 1466 |         13 => "Out of Calibration", | support logic; impact depends on neighboring block and data flow. |
| 1467 |         14 => "Special Instructions", | support logic; impact depends on neighboring block and data flow. |
| 1468 |         15 => "Above Normal — Least Severe", | support logic; impact depends on neighboring block and data flow
| 1469 |         16 => "Above Normal — Moderately Severe", | support logic; impact depends on neighboring block and data
| 1470 |         17 => "Below Normal — Least Severe", | support logic; impact depends on neighboring block and data flow
| 1471 |         18 => "Below Normal — Moderately Severe", | support logic; impact depends on neighboring block and data
| 1472 |         19 => "Received Network Data in Error", | support logic; impact depends on neighboring block and data f
| 1473 |         31 => "Condition Exists", | support logic; impact depends on neighboring block and data flow. |
| 1474 |         _ => "Reserved", | support logic; impact depends on neighboring block and data flow. |
| 1475 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1476 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1477 | (blank) | blank separator; no runtime behavior. |
| 1478 | // ■■ J1939 Bus Logger ■■■
```

```
| 1524 |             // 500 kbps, min frame ~55 bits → ~9090 frames/s max | comment/documentation; changing affects und
| 1525 |             self.bus_load_pct = (self.frame_count as f64 / 9090.0 * 100.0).min(100.0); | support logic; impact
| 1526 |             self.frame_count = 0; | support logic; impact depends on neighboring block and data flow. |
| 1527 |             self.frame_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1528 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1529 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1530 | (blank) | blank separator; no runtime behavior. |
| 1531 |     pub fn visible_frames(&self) -> impl Iterator<Item = &J1939Frame> { | function signature/body; changes affe
| 1532 |         self.frames.iter().filter(\|f\| { | support logic; impact depends on neighboring block and data flow.
| 1533 |             self.filter_pgn.is_none_or(\|p\| f.pgn == p) | support logic; impact depends on neighboring block a
| 1534 |                 && self.filter_sa.is_none_or(\|s\| f.sa == s) | support logic; impact depends on neighboring bl
| 1535 |         }) | support logic; impact depends on neighboring block and data flow. |
| 1536 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1537 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1538 | (blank) | blank separator; no runtime behavior. |
| 1539 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 1540 | mod tests { | module declaration; changing can break namespace graph. |
| 1541 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 1542 | (blank) | blank separator; no runtime behavior. |
| 1543 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1544 |     fn pdu1_id_roundtrip_preserves_destination() { | function signature/body; changes affect API behavior and c
| 1545 |         let id = J1939Frame::build_id(3, pgn::TSC1, addr::ECM_1, addr::TRANSMISSION); | support logic; impact c
| 1546 |         let frame = J1939Frame::from_raw(0.0, id, &[0xFF; 8]); | support logic; impact depends on neighboring b
| 1547 |         assert_eq!(frame.pgn, pgn::TSC1); | support logic; impact depends on neighboring block and data flow.
| 1548 |         assert_eq!(frame.da, addr::TRANSMISSION); | support logic; impact depends on neighboring block and data
| 1549 |         assert_eq!(frame.sa, addr::ECM_1); | support logic; impact depends on neighboring block and data flow.
| 1550 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1551 | (blank) | blank separator; no runtime behavior. |
| 1552 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1553 |     fn ebc2_decodes_all_wheel_speeds() { | function signature/body; changes affect API behavior and control-flo
| 1554 |         let mut data = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1555 |         let fl = (48.0 / 0.125) as u16; | support logic; impact depends on neighboring block and data flow. |
| 1556 |         let fr = (50.0 / 0.125) as u16; | support logic; impact depends on neighboring block and data flow. |
| 1557 |         let rl = (47.0 / 0.125) as u16; | support logic; impact depends on neighboring block and data flow. |
| 1558 |         let rr = (49.0 / 0.125) as u16; | support logic; impact depends on neighboring block and data flow. |
| 1559 |         data[0] = (fl & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1560 |         data[1] = (fl >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1561 |         data[2] = (fr & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1562 |         data[3] = (fr >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1563 |         data[4] = (rl & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1564 |         data[5] = (rl >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1565 |         data[6] = (rr & 0xFF) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1566 |         data[7] = (rr >> 8) as u8; | support logic; impact depends on neighboring block and data flow. |
| 1567 |         let id = J1939Frame::build_id(6, pgn::EBC2, addr::BRAKES, 0xFF); | support logic; impact depends on nei
| 1568 |         let frame = J1939Frame::from_raw(0.0, id, &data); | support logic; impact depends on neighboring block
| 1569 |         assert_eq!(frame.pgn_name(), "EBC2"); | support logic; impact depends on neighboring block and data flo
| 1570 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1571 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1572 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1573 |             .any(\|d\| d.spn == 904 && (d.physical - 48.0).abs() < 0.2)); | support logic; impact depends on ne
| 1574 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1575 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1576 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1577 |             .any(\|d\| d.spn == 905 && (d.physical - 50.0).abs() < 0.2)); | support logic; impact depends on ne
| 1578 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1579 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1580 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1581 |             .any(\|d\| d.spn == 906 && (d.physical - 47.0).abs() < 0.2)); | support logic; impact depends on ne
| 1582 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1583 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1584 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1585 |             .any(\|d\| d.spn == 907 && (d.physical - 49.0).abs() < 0.2)); | support logic; impact depends on ne
| 1586 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1587 | (blank) | blank separator; no runtime behavior. |
| 1588 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1589 |     fn etc2_gear_decode_matches_encoded_value() { | function signature/body; changes affect API behavior and co
| 1590 |         let mut data = [0xFFu8; 8]; | support logic; impact depends on neighboring block and data flow. |
| 1591 |         data[3] = 132; // selected gear = 7 after offset -125 | support logic; impact depends on neighboring bl
| 1592 |         data[4] = 132; // current gear = 7 | support logic; impact depends on neighboring block and data flow.
| 1593 |         data[7] = 3; // range C | support logic; impact depends on neighboring block and data flow. |
| 1594 |         let id = J1939Frame::build_id(3, pgn::ETC2, addr::TRANSMISSION, 0xFF); | support logic; impact depends
| 1595 |         let frame = J1939Frame::from_raw(0.0, id, &data); | support logic; impact depends on neighboring block
| 1596 |         assert_eq!(frame.pgn_name(), "ETC2"); | support logic; impact depends on neighboring block and data flo
| 1597 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1598 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1599 |             .iter() | support logic; impact depends on neighboring block and data flow. |
```

```
| 1600 |             .any(\|d\| d.spn == 524 && (d.physical - 7.0).abs() < 0.1)); | support logic; impact depends on nei
| 1601 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1602 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1603 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1604 |             .any(\|d\| d.spn == 523 && (d.physical - 7.0).abs() < 0.1)); | support logic; impact depends on nei
| 1605 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1606 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1607 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1608 |             .any(\|d\| d.spn == 525 && (d.physical - 3.0).abs() < 0.1)); | support logic; impact depends on nei
| 1609 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1610 | (blank) | blank separator; no runtime behavior. |
| 1611 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1612 |     fn adas_builder_roundtrip_exposes_core_spns() { | function signature/body; changes affect API behavior and
| 1613 |         let frame = Builder::adas1(0.0, true, true, false, 23.4, 1.85, 80.0, addr::HEADWAY); | support logic; i
| 1614 |         assert_eq!(frame.pgn, pgn::ADAS1); | support logic; impact depends on neighboring block and data flow.
| 1615 |         assert_eq!(frame.pgn_name(), "ADAS1"); | support logic; impact depends on neighboring block and data fl
| 1616 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1617 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1618 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1619 |             .any(\|d\| d.spn == 52001 && d.physical >= 1.0)); | support logic; impact depends on neighboring bl
| 1620 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1621 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1622 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1623 |             .any(\|d\| d.spn == 52002 && d.physical >= 1.0)); | support logic; impact depends on neighboring bl
| 1624 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1625 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1626 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1627 |             .any(\|d\| d.spn == 52004 && (d.physical - 23.4).abs() < 0.2)); | support logic; impact depends on
| 1628 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1629 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1630 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1631 |             .any(\|d\| d.spn == 52005 && (d.physical - 1.85).abs() < 0.05)); | support logic; impact depends on
| 1632 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1633 | (blank) | blank separator; no runtime behavior. |
| 1634 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1635 |     fn energy_builder_roundtrip_exposes_power_values() { | function signature/body; changes affect API behavior
| 1636 |         let frame = Builder::engy1(0.0, 27.4, 84.0, 2.30, 19.75, addr::HEADWAY); | support logic; impact depend
| 1637 |         assert_eq!(frame.pgn, pgn::ENGY1); | support logic; impact depends on neighboring block and data flow.
| 1638 |         assert_eq!(frame.pgn_name(), "ENGY1"); | support logic; impact depends on neighboring block and data fl
| 1639 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1640 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1641 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1642 |             .any(\|d\| d.spn == 52201 && (d.physical - 27.4).abs() < 0.2)); | support logic; impact depends on
| 1643 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1644 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1645 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1646 |             .any(\|d\| d.spn == 52202 && (d.physical - 84.0).abs() < 1.0)); | support logic; impact depends on
| 1647 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1648 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1649 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1650 |             .any(\|d\| d.spn == 52203 && (d.physical - 2.30).abs() < 0.05)); | support logic; impact depends on
| 1651 |         assert!(frame | support logic; impact depends on neighboring block and data flow. |
| 1652 |             .decoded | support logic; impact depends on neighboring block and data flow. |
| 1653 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1654 |             .any(\|d\| d.spn == 52204 && (d.physical - 19.75).abs() < 0.05)); | support logic; impact depends o
| 1655 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1656 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/leak_physics.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 2151 | Non-empty: 1994 | Symbol count: 95

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 29 |
| len | 14 |
| is_empty | 12 |
| push | 11 |
| step | 11 |
| with_default_machine_presets | 9 |
```

```
| runtime_profile | 8 |
| reset | 7 |
| name | 5 |
| density_kg_m3 | 4 |
| engineering_material_catalog | 4 |
| engineering_oil_catalog | 4 |
| apply_manual_params | 4 |
| default | 3 |
| temp_window_c | 3 |
| circuit_mut | 3 |
| predict_scenarios | 3 |
| fmt | 2 |
| all | 2 |
| viscosity_index | 2 |
| gas_permeability_factor | 2 |
| update | 2 |
| aggregate_group_report | 2 |
| run_monte_carlo | 2 |
| add_circuit | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | struct | LeakModelCoefficients | data/schema coupling and match/serde break risk |
| 16 | impl | Default for LeakModelCoefficients | method behavior and trait semantics |
| 17 | fn | default | API and behavior coupling to callers/implementers |
| 29 | struct | CalibrationCsvSample | data/schema coupling and match/serde break risk |
| 43 | struct | CalibrationCircuitReport | data/schema coupling and match/serde break risk |
| 56 | struct | CalibrationGroupReport | data/schema coupling and match/serde break risk |
| 65 | struct | CalibrationReport | data/schema coupling and match/serde break risk |
| 73 | fn | aggregate_group_report | API and behavior coupling to callers/implementers |
| 107 | enum | LeakAlertLevel | data/schema coupling and match/serde break risk |
| 116 | enum | PressureBand | data/schema coupling and match/serde break risk |
| 125 | impl | std::fmt::Display for PressureBand | method behavior and trait semantics |
| 126 | fn | fmt | API and behavior coupling to callers/implementers |
| 140 | enum | CircuitComponent | data/schema coupling and match/serde break risk |
| 147 | enum | OilType | data/schema coupling and match/serde break risk |
| 195 | struct | OilRuntimeProfile | data/schema coupling and match/serde break risk |
| 203 | const | OIL_TYPE_VARIANTS | namespace and compile-time linkage |
| 248 | impl | OilType | method behavior and trait semantics |
| 249 | fn | all | API and behavior coupling to callers/implementers |
| 253 | fn | name | API and behavior coupling to callers/implementers |
| 257 | fn | runtime_profile | API and behavior coupling to callers/implementers |
| 307 | impl | OilType | method behavior and trait semantics |
| 308 | fn | density_kg_m3 | API and behavior coupling to callers/implementers |
| 312 | fn | viscosity_index | API and behavior coupling to callers/implementers |
| 317 | impl | std::fmt::Display for LeakAlertLevel | method behavior and trait semantics |
| 318 | fn | fmt | API and behavior coupling to callers/implementers |
| 331 | enum | OringMaterial | data/schema coupling and match/serde break risk |
| 351 | struct | MaterialRuntimeProfile | data/schema coupling and match/serde break risk |
| 358 | const | ORING_MATERIAL_VARIANTS | namespace and compile-time linkage |
| 373 | impl | OringMaterial | method behavior and trait semantics |
| 374 | fn | all | API and behavior coupling to callers/implementers |
| 378 | fn | name | API and behavior coupling to callers/implementers |
| 382 | fn | runtime_profile | API and behavior coupling to callers/implementers |
| 403 | fn | temp_window_c | API and behavior coupling to callers/implementers |
| 408 | fn | gas_permeability_factor | API and behavior coupling to callers/implementers |
| 414 | struct | MaterialCatalogEntry | data/schema coupling and match/serde break risk |
| 435 | struct | OilCatalogEntry | data/schema coupling and match/serde break risk |
| 458 | fn | engineering_material_catalog | API and behavior coupling to callers/implementers |
| 496 | fn | engineering_oil_catalog | API and behavior coupling to callers/implementers |
| 540 | struct | PressureEnvelope | data/schema coupling and match/serde break risk |
| 549 | struct | OringSpec | data/schema coupling and match/serde break risk |
| 563 | struct | CircuitInput | data/schema coupling and match/serde break risk |
| 573 | struct | CircuitResult | data/schema coupling and match/serde break risk |
| 599 | struct | ManualCircuitParams | data/schema coupling and match/serde break risk |
| 615 | struct | ScenarioPrediction | data/schema coupling and match/serde break risk |
| 626 | struct | LeakAlert | data/schema coupling and match/serde break risk |
| 634 | struct | PressureStats | data/schema coupling and match/serde break risk |
| 640 | impl | PressureStats | method behavior and trait semantics |
| 641 | fn | new | API and behavior coupling to callers/implementers |
| 649 | fn | update | API and behavior coupling to callers/implementers |
| 657 | struct | SealState | data/schema coupling and match/serde break risk |
| 671 | impl | SealState | method behavior and trait semantics |
| 672 | fn | new | API and behavior coupling to callers/implementers |
```

```
| 690 | struct | LeakCircuit | data/schema coupling and match/serde break risk |
| 707 | impl | LeakCircuit | method behavior and trait semantics |
| 709 | fn | new | API and behavior coupling to callers/implementers |
| 740 | fn | step | API and behavior coupling to callers/implementers |
| 1011 | fn | reset | API and behavior coupling to callers/implementers |
| 1016 | fn | apply_manual_params | API and behavior coupling to callers/implementers |
| 1062 | struct | LeakPhysicsRig | data/schema coupling and match/serde break risk |
| 1069 | impl | Default for LeakPhysicsRig | method behavior and trait semantics |
| 1070 | fn | default | API and behavior coupling to callers/implementers |
| 1075 | impl | LeakPhysicsRig | method behavior and trait semantics |
| 1076 | fn | new | API and behavior coupling to callers/implementers |
| 1085 | fn | with_default_machine_presets | API and behavior coupling to callers/implementers |
| 1184 | fn | add_circuit | API and behavior coupling to callers/implementers |
| 1188 | fn | circuit_mut | API and behavior coupling to callers/implementers |
| 1192 | fn | oil_density_for | API and behavior coupling to callers/implementers |
| 1200 | fn | set_runtime_pressure_state | API and behavior coupling to callers/implementers |
| 1212 | fn | apply_manual_params | API and behavior coupling to callers/implementers |
| 1221 | fn | step | API and behavior coupling to callers/implementers |
| 1246 | fn | reset | API and behavior coupling to callers/implementers |
| 1255 | fn | export_last_results_json | API and behavior coupling to callers/implementers |
| 1267 | fn | export_last_results_csv | API and behavior coupling to callers/implementers |
| 1316 | fn | export_predictions_json | API and behavior coupling to callers/implementers |
| 1332 | fn | export_predictions_csv | API and behavior coupling to callers/implementers |
| 1362 | fn | export_material_catalog_json | API and behavior coupling to callers/implementers |
| 1374 | fn | export_material_catalog_csv | API and behavior coupling to callers/implementers |
| 1408 | fn | export_oil_catalog_json | API and behavior coupling to callers/implementers |
| 1420 | fn | export_oil_catalog_csv | API and behavior coupling to callers/implementers |
| 1457 | fn | calibrate_from_csv | API and behavior coupling to callers/implementers |
| 1462 | struct | CsvRow | data/schema coupling and match/serde break risk |
| 1654 | fn | export_calibration_report_json | API and behavior coupling to callers/implementers |
| 1670 | fn | export_calibration_report_csv | API and behavior coupling to callers/implementers |
| 1725 | fn | run_monte_carlo | API and behavior coupling to callers/implementers |
| 1816 | fn | predict_scenarios | API and behavior coupling to callers/implementers |
| 1908 | module | tests | namespace and compile-time linkage |
| 1912 | fn | high_pressure_eventually_ruptures | API and behavior coupling to callers/implementers |
| 1944 | fn | manual_params_are_applied | API and behavior coupling to callers/implementers |
| 1975 | fn | scenario_prediction_returns_ranked_data | API and behavior coupling to callers/implementers |
| 1983 | fn | monte_carlo_produces_samples | API and behavior coupling to callers/implementers |
| 1990 | fn | exports_json_and_csv_files | API and behavior coupling to callers/implementers |
| 2030 | fn | report_captures_pressure_band_and_safe_window | API and behavior coupling to callers/implementers |
| 2055 | fn | rupture_event_records_pressure_and_time | API and behavior coupling to callers/implementers |
| 2089 | fn | engineering_catalogs_are_large_and_coherent | API and behavior coupling to callers/implementers |
| 2107 | fn | calibration_mode_ingests_csv_and_exports_reports | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! O-ring leak / rupture physics for hydraulic, oil and refrigerant circuits. | comment/documentation; changing a
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | use serde::{Deserialize, Serialize}; | import wiring; changing path/name can break compilation and module linkage
| 4 | use std::fs; | import wiring; changing path/name can break compilation and module linkage. |
| 5 | use std::path::Path; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | #[derive(Debug, Clone, Copy, Serialize, Deserialize)] | comment/documentation; changing affects understanding and
| 8 | pub struct LeakModelCoefficients { | data layout contract; field changes can break callers/serde/tests. |
| 9 |     pub damage_rate_scale: f64, | support logic; impact depends on neighboring block and data flow. |
| 10 |     pub extrusion_rate_scale: f64, | support logic; impact depends on neighboring block and data flow. |
| 11 |     pub thermal_rate_scale: f64, | support logic; impact depends on neighboring block and data flow. |
| 12 |     pub flow_rate_scale: f64, | support logic; impact depends on neighboring block and data flow. |
| 13 |     pub rupture_area_scale: f64, | support logic; impact depends on neighboring block and data flow. |
| 14 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 15 | (blank) | blank separator; no runtime behavior. |
| 16 | impl Default for LeakModelCoefficients { | implementation binding; behavior semantics and trait conformance live
| 17 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 18 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 19 |             damage_rate_scale: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 20 |             extrusion_rate_scale: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 21 |             thermal_rate_scale: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 22 |             flow_rate_scale: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 23 |             rupture_area_scale: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 24 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 25 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 26 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 | (blank) | blank separator; no runtime behavior. |
| 28 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
```

```
| 29 | pub struct CalibrationCsvSample { | data layout contract; field changes can break callers/serde/tests. |
| 30 |     pub timestamp_s: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 31 |     pub circuit_name: String, | support logic; impact depends on neighboring block and data flow. |
| 32 |     pub pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 33 |     pub delta_p_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 34 |     pub temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 35 |     pub cycles_per_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 36 |     pub duty_01: f64, | support logic; impact depends on neighboring block and data flow. |
| 37 |     pub fluid_density_kg_m3: f64, | support logic; impact depends on neighboring block and data flow. |
| 38 |     pub measured_leak_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 39 |     pub observed_rupture: Option<bool>, | support logic; impact depends on neighboring block and data flow. |
| 40 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 41 | (blank) | blank separator; no runtime behavior. |
| 42 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
| 43 | pub struct CalibrationCircuitReport { | data layout contract; field changes can break callers/serde/tests. |
| 44 |     pub circuit_name: String, | support logic; impact depends on neighboring block and data flow. |
| 45 |     pub material: String, | support logic; impact depends on neighboring block and data flow. |
| 46 |     pub oil: String, | support logic; impact depends on neighboring block and data flow. |
| 47 |     pub sample_count: usize, | support logic; impact depends on neighboring block and data flow. |
| 48 |     pub rmse_leak_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 49 |     pub mape_leak_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 50 |     pub max_abs_error_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 51 |     pub rupture_accuracy_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 52 |     pub fitted: LeakModelCoefficients, | support logic; impact depends on neighboring block and data flow. |
| 53 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 | (blank) | blank separator; no runtime behavior. |
| 55 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
| 56 | pub struct CalibrationGroupReport { | data layout contract; field changes can break callers/serde/tests. |
| 57 |     pub group: String, | support logic; impact depends on neighboring block and data flow. |
| 58 |     pub sample_count: usize, | support logic; impact depends on neighboring block and data flow. |
| 59 |     pub mean_rmse_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 60 |     pub mean_mape_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 61 |     pub mean_rupture_accuracy_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 62 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 63 | (blank) | blank separator; no runtime behavior. |
| 64 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inter
| 65 | pub struct CalibrationReport { | data layout contract; field changes can break callers/serde/tests. |
| 66 |     pub total_samples: usize, | support logic; impact depends on neighboring block and data flow. |
| 67 |     pub calibrated_circuits: usize, | support logic; impact depends on neighboring block and data flow. |
| 68 |     pub circuit_reports: Vec<CalibrationCircuitReport>, | support logic; impact depends on neighboring block and
| 69 |     pub by_material: Vec<CalibrationGroupReport>, | support logic; impact depends on neighboring block and data
| 70 |     pub by_oil: Vec<CalibrationGroupReport>, | support logic; impact depends on neighboring block and data flow.
| 71 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 72 | (blank) | blank separator; no runtime behavior. |
| 73 | fn aggregate_group_report<F>( | function signature/body; changes affect API behavior and control-flow. |
| 74 |     items: &[CalibrationCircuitReport], | support logic; impact depends on neighboring block and data flow. |
| 75 |     mut key_fn: F, | support logic; impact depends on neighboring block and data flow. |
| 76 | ) -> Vec<CalibrationGroupReport> | support logic; impact depends on neighboring block and data flow. |
| 77 | where | support logic; impact depends on neighboring block and data flow. |
| 78 |     F: FnMut(&CalibrationCircuitReport) -> String, | support logic; impact depends on neighboring block and data
| 79 | { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 80 |     use std::collections::BTreeMap; | import wiring; changing path/name can break compilation and module linkage
| 81 |     let mut acc: BTreeMap<String, (usize, f64, f64, f64, usize)> = BTreeMap::new(); | support logic; impact depe
| 82 |     for r in items { | iteration logic; may alter timing, throughput, and safety constraints. |
| 83 |         let key = key_fn(r); | support logic; impact depends on neighboring block and data flow. |
| 84 |         let entry = acc.entry(key).or_insert((0, 0.0, 0.0, 0.0, 0)); | support logic; impact depends on neighbor
| 85 |         entry.0 += 1; | support logic; impact depends on neighboring block and data flow. |
| 86 |         entry.1 += r.rmse_leak_lpm; | support logic; impact depends on neighboring block and data flow. |
| 87 |         entry.2 += r.mape_leak_pct; | support logic; impact depends on neighboring block and data flow. |
| 88 |         entry.3 += r.rupture_accuracy_pct; | support logic; impact depends on neighboring block and data flow. |
| 89 |         entry.4 += r.sample_count; | support logic; impact depends on neighboring block and data flow. |
| 90 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 91 | (blank) | blank separator; no runtime behavior. |
| 92 |     let mut out = Vec::with_capacity(acc.len()); | support logic; impact depends on neighboring block and data f
| 93 |     for (group, (count, rmse, mape, rupture_acc, sample_count)) in acc { | iteration logic; may alter timing, th
| 94 |         let n = count.max(1) as f64; | support logic; impact depends on neighboring block and data flow. |
| 95 |         out.push(CalibrationGroupReport { | support logic; impact depends on neighboring block and data flow. |
| 96 |             group, | support logic; impact depends on neighboring block and data flow. |
| 97 |             sample_count, | support logic; impact depends on neighboring block and data flow. |
| 98 |             mean_rmse_lpm: rmse / n, | support logic; impact depends on neighboring block and data flow. |
| 99 |             mean_mape_pct: mape / n, | support logic; impact depends on neighboring block and data flow. |
| 100 |             mean_rupture_accuracy_pct: rupture_acc / n, | support logic; impact depends on neighboring block an
| 101 |         }); | support logic; impact depends on neighboring block and data flow. |
| 102 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 103 |     out | support logic; impact depends on neighboring block and data flow. |
| 104 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 105 | (blank) | blank separator; no runtime behavior. |
| 106 | #[derive(Debug, Clone, Copy, PartialEq, Eq, PartialOrd, Ord, Serialize, Deserialize)] | comment/documentation; c
| 107 | pub enum LeakAlertLevel { | state machine variants; changes ripple through matches and business logic. |
| 108 |     Normal, | support logic; impact depends on neighboring block and data flow. |
| 109 |     Watch, | support logic; impact depends on neighboring block and data flow. |
| 110 |     Warning, | support logic; impact depends on neighboring block and data flow. |
| 111 |     Critical, | support logic; impact depends on neighboring block and data flow. |
| 112 |     Ruptured, | support logic; impact depends on neighboring block and data flow. |
| 113 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 114 | (blank) | blank separator; no runtime behavior. |
| 115 | #[derive(Debug, Clone, Copy, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects c
| 116 | pub enum PressureBand { | state machine variants; changes ripple through matches and business logic. |
| 117 |     BelowMinimum, | support logic; impact depends on neighboring block and data flow. |
| 118 |     LowBand, | support logic; impact depends on neighboring block and data flow. |
| 119 |     MidBand, | support logic; impact depends on neighboring block and data flow. |
| 120 |     HighBand, | support logic; impact depends on neighboring block and data flow. |
| 121 |     OverMaximum, | support logic; impact depends on neighboring block and data flow. |
| 122 |     AtRupture, | support logic; impact depends on neighboring block and data flow. |
| 123 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 124 | (blank) | blank separator; no runtime behavior. |
| 125 | impl std::fmt::Display for PressureBand { | implementation binding; behavior semantics and trait conformance liv
| 126 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 127 |         let s = match self { | support logic; impact depends on neighboring block and data flow. |
| 128 |             PressureBand::BelowMinimum => "below_min", | support logic; impact depends on neighboring block and
| 129 |             PressureBand::LowBand => "low", | support logic; impact depends on neighboring block and data flow.
| 130 |             PressureBand::MidBand => "mid", | support logic; impact depends on neighboring block and data flow.
| 131 |             PressureBand::HighBand => "high", | support logic; impact depends on neighboring block and data flow
| 132 |             PressureBand::OverMaximum => "over_max", | support logic; impact depends on neighboring block and da
| 133 |             PressureBand::AtRupture => "at_rupture", | support logic; impact depends on neighboring block and da
| 134 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 135 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 136 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 137 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 | (blank) | blank separator; no runtime behavior. |
| 139 | #[derive(Debug, Clone, Copy, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects c
| 140 | pub enum CircuitComponent { | state machine variants; changes ripple through matches and business logic. |
| 141 |     Oring, | support logic; impact depends on neighboring block and data flow. |
| 142 |     Seal, | support logic; impact depends on neighboring block and data flow. |
| 143 |     AcHose, | support logic; impact depends on neighboring block and data flow. |
| 144 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 145 | (blank) | blank separator; no runtime behavior. |
| 146 | #[derive(Debug, Clone, Copy, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects c
| 147 | pub enum OilType { | state machine variants; changes ripple through matches and business logic. |
| 148 |     HydraulicIso22, | support logic; impact depends on neighboring block and data flow. |
| 149 |     HydraulicIso32, | support logic; impact depends on neighboring block and data flow. |
| 150 |     HydraulicIso46, | support logic; impact depends on neighboring block and data flow. |
| 151 |     HydraulicIso68, | support logic; impact depends on neighboring block and data flow. |
| 152 |     HydraulicIso100, | support logic; impact depends on neighboring block and data flow. |
| 153 |     HydraulicIso150, | support logic; impact depends on neighboring block and data flow. |
| 154 |     HydraulicIso220, | support logic; impact depends on neighboring block and data flow. |
| 155 |     Aw32, | support logic; impact depends on neighboring block and data flow. |
| 156 |     Aw46, | support logic; impact depends on neighboring block and data flow. |
| 157 |     Aw68, | support logic; impact depends on neighboring block and data flow. |
| 158 |     Utto10w30, | support logic; impact depends on neighboring block and data flow. |
| 159 |     Utto80w, | support logic; impact depends on neighboring block and data flow. |
| 160 |     Gear75w90, | support logic; impact depends on neighboring block and data flow. |
| 161 |     Gear80w90, | support logic; impact depends on neighboring block and data flow. |
| 162 |     AtfDexronIii, | support logic; impact depends on neighboring block and data flow. |
| 163 |     AtfDexronVi, | support logic; impact depends on neighboring block and data flow. |
| 164 |     Engine0w20, | support logic; impact depends on neighboring block and data flow. |
| 165 |     Engine5w30, | support logic; impact depends on neighboring block and data flow. |
| 166 |     Engine5w40, | support logic; impact depends on neighboring block and data flow. |
| 167 |     Engine10w30, | support logic; impact depends on neighboring block and data flow. |
| 168 |     Engine15w40, | support logic; impact depends on neighboring block and data flow. |
| 169 |     Engine20w50, | support logic; impact depends on neighboring block and data flow. |
| 170 |     TransformerInhibited, | support logic; impact depends on neighboring block and data flow. |
| 171 |     Turbine32, | support logic; impact depends on neighboring block and data flow. |
| 172 |     Turbine46, | support logic; impact depends on neighboring block and data flow. |
| 173 |     Compressor46, | support logic; impact depends on neighboring block and data flow. |
| 174 |     Compressor68, | support logic; impact depends on neighboring block and data flow. |
| 175 |     BiodegradableHees46, | support logic; impact depends on neighboring block and data flow. |
| 176 |     BiodegradableHepr46, | support logic; impact depends on neighboring block and data flow. |
| 177 |     FireResistantHfdu46, | support logic; impact depends on neighboring block and data flow. |
| 178 |     BrakeDot3, | support logic; impact depends on neighboring block and data flow. |
| 179 |     BrakeDot4, | support logic; impact depends on neighboring block and data flow. |
| 180 |     BrakeDot51, | support logic; impact depends on neighboring block and data flow. |
```

```
| 181 |     SyntheticEsters100, | support logic; impact depends on neighboring block and data flow. |
| 182 |     RefrigerationMineral3gs, | support logic; impact depends on neighboring block and data flow. |
| 183 |     Pag46, | support logic; impact depends on neighboring block and data flow. |
| 184 |     Pag100, | support logic; impact depends on neighboring block and data flow. |
| 185 |     Pag150, | support logic; impact depends on neighboring block and data flow. |
| 186 |     Poe32, | support logic; impact depends on neighboring block and data flow. |
| 187 |     Poe46, | support logic; impact depends on neighboring block and data flow. |
| 188 |     Poe68, | support logic; impact depends on neighboring block and data flow. |
| 189 |     Poe100, | support logic; impact depends on neighboring block and data flow. |
| 190 |     VacuumPump100, | support logic; impact depends on neighboring block and data flow. |
| 191 |     Custom, | support logic; impact depends on neighboring block and data flow. |
| 192 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 193 | (blank) | blank separator; no runtime behavior. |
| 194 | #[derive(Debug, Clone, Copy, Serialize, Deserialize)] | comment/documentation; changing affects understanding an
| 195 | pub struct OilRuntimeProfile { | data layout contract; field changes can break callers/serde/tests. |
| 196 |     pub label: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 197 |     pub density_kg_m3: f64, | support logic; impact depends on neighboring block and data flow. |
| 198 |     pub viscosity_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 199 |     pub min_oper_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 200 |     pub max_oper_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 201 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 202 | (blank) | blank separator; no runtime behavior. |
| 203 | const OIL_TYPE_VARIANTS: [OilType; 42] = [ | support logic; impact depends on neighboring block and data flow.
| 204 |     OilType::HydraulicIso22, | support logic; impact depends on neighboring block and data flow. |
| 205 |     OilType::HydraulicIso32, | support logic; impact depends on neighboring block and data flow. |
| 206 |     OilType::HydraulicIso46, | support logic; impact depends on neighboring block and data flow. |
| 207 |     OilType::HydraulicIso68, | support logic; impact depends on neighboring block and data flow. |
| 208 |     OilType::HydraulicIso100, | support logic; impact depends on neighboring block and data flow. |
| 209 |     OilType::HydraulicIso150, | support logic; impact depends on neighboring block and data flow. |
| 210 |     OilType::HydraulicIso220, | support logic; impact depends on neighboring block and data flow. |
| 211 |     OilType::Aw32, | support logic; impact depends on neighboring block and data flow. |
| 212 |     OilType::Aw46, | support logic; impact depends on neighboring block and data flow. |
| 213 |     OilType::Aw68, | support logic; impact depends on neighboring block and data flow. |
| 214 |     OilType::Utto10w30, | support logic; impact depends on neighboring block and data flow. |
| 215 |     OilType::Utto80w, | support logic; impact depends on neighboring block and data flow. |
| 216 |     OilType::Gear75w90, | support logic; impact depends on neighboring block and data flow. |
| 217 |     OilType::Gear80w90, | support logic; impact depends on neighboring block and data flow. |
| 218 |     OilType::AtfDexronIii, | support logic; impact depends on neighboring block and data flow. |
| 219 |     OilType::AtfDexronVi, | support logic; impact depends on neighboring block and data flow. |
| 220 |     OilType::Engine0w20, | support logic; impact depends on neighboring block and data flow. |
| 221 |     OilType::Engine5w30, | support logic; impact depends on neighboring block and data flow. |
| 222 |     OilType::Engine5w40, | support logic; impact depends on neighboring block and data flow. |
| 223 |     OilType::Engine10w30, | support logic; impact depends on neighboring block and data flow. |
| 224 |     OilType::Engine15w40, | support logic; impact depends on neighboring block and data flow. |
| 225 |     OilType::Engine20w50, | support logic; impact depends on neighboring block and data flow. |
| 226 |     OilType::TransformerInhibited, | support logic; impact depends on neighboring block and data flow. |
| 227 |     OilType::Turbine32, | support logic; impact depends on neighboring block and data flow. |
| 228 |     OilType::Turbine46, | support logic; impact depends on neighboring block and data flow. |
| 229 |     OilType::Compressor46, | support logic; impact depends on neighboring block and data flow. |
| 230 |     OilType::Compressor68, | support logic; impact depends on neighboring block and data flow. |
| 231 |     OilType::BiodegradableHees46, | support logic; impact depends on neighboring block and data flow. |
| 232 |     OilType::BiodegradableHepr46, | support logic; impact depends on neighboring block and data flow. |
| 233 |     OilType::FireResistantHfdu46, | support logic; impact depends on neighboring block and data flow. |
| 234 |     OilType::BrakeDot3, | support logic; impact depends on neighboring block and data flow. |
| 235 |     OilType::BrakeDot4, | support logic; impact depends on neighboring block and data flow. |
| 236 |     OilType::BrakeDot51, | support logic; impact depends on neighboring block and data flow. |
| 237 |     OilType::SyntheticEsters100, | support logic; impact depends on neighboring block and data flow. |
| 238 |     OilType::RefrigerationMineral3gs, | support logic; impact depends on neighboring block and data flow. |
| 239 |     OilType::Pag46, | support logic; impact depends on neighboring block and data flow. |
| 240 |     OilType::Pag100, | support logic; impact depends on neighboring block and data flow. |
| 241 |     OilType::Pag150, | support logic; impact depends on neighboring block and data flow. |
| 242 |     OilType::Poe32, | support logic; impact depends on neighboring block and data flow. |
| 243 |     OilType::Poe46, | support logic; impact depends on neighboring block and data flow. |
| 244 |     OilType::Poe68, | support logic; impact depends on neighboring block and data flow. |
| 245 |     OilType::Poe100, | support logic; impact depends on neighboring block and data flow. |
| 246 | ]; | support logic; impact depends on neighboring block and data flow. |
| 247 | (blank) | blank separator; no runtime behavior. |
| 248 | impl OilType { | implementation binding; behavior semantics and trait conformance live here. |
| 249 |     pub fn all() -> &'static [OilType] { | function signature/body; changes affect API behavior and control-flow
| 250 |         &OIL_TYPE_VARIANTS | support logic; impact depends on neighboring block and data flow. |
| 251 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 252 | (blank) | blank separator; no runtime behavior. |
| 253 |     pub fn name(self) -> &'static str { | function signature/body; changes affect API behavior and control-flow
| 254 |         self.runtime_profile().label | support logic; impact depends on neighboring block and data flow. |
| 255 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 256 | (blank) | blank separator; no runtime behavior. |
```

```
| 257 |     pub fn runtime_profile(self) -> OilRuntimeProfile { | function signature/body; changes affect API behavior a
| 258 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 259 |             OilType::HydraulicIso22 => OilRuntimeProfile { label: "Hydraulic ISO VG 22", density_kg_m3: 852.0, v
| 260 |             OilType::HydraulicIso32 => OilRuntimeProfile { label: "Hydraulic ISO VG 32", density_kg_m3: 858.0, v
| 261 |             OilType::HydraulicIso46 => OilRuntimeProfile { label: "Hydraulic ISO VG 46", density_kg_m3: 865.0, v
| 262 |             OilType::HydraulicIso68 => OilRuntimeProfile { label: "Hydraulic ISO VG 68", density_kg_m3: 875.0, v
| 263 |             OilType::HydraulicIso100 => OilRuntimeProfile { label: "Hydraulic ISO VG 100", density_kg_m3: 882.0,
| 264 |             OilType::HydraulicIso150 => OilRuntimeProfile { label: "Hydraulic ISO VG 150", density_kg_m3: 890.0,
| 265 |             OilType::HydraulicIso220 => OilRuntimeProfile { label: "Hydraulic ISO VG 220", density_kg_m3: 896.0,
| 266 |             OilType::Aw32 => OilRuntimeProfile { label: "AW Hydraulic 32", density_kg_m3: 860.0, viscosity_index
| 267 |             OilType::Aw46 => OilRuntimeProfile { label: "AW Hydraulic 46", density_kg_m3: 868.0, viscosity_index
| 268 |             OilType::Aw68 => OilRuntimeProfile { label: "AW Hydraulic 68", density_kg_m3: 878.0, viscosity_index
| 269 |             OilType::Utto10w30 => OilRuntimeProfile { label: "UTTO 10W-30", density_kg_m3: 872.0, viscosity_inde
| 270 |             OilType::Utto80w => OilRuntimeProfile { label: "UTTO 80W", density_kg_m3: 885.0, viscosity_index: 12
| 271 |             OilType::Gear75w90 => OilRuntimeProfile { label: "Gear Oil 75W-90", density_kg_m3: 875.0, viscosity_
| 272 |             OilType::Gear80w90 => OilRuntimeProfile { label: "Gear Oil 80W-90", density_kg_m3: 890.0, viscosity_
| 273 |             OilType::AtfDexronIii => OilRuntimeProfile { label: "ATF Dexron III", density_kg_m3: 850.0, viscosit
| 274 |             OilType::AtfDexronVi => OilRuntimeProfile { label: "ATF Dexron VI", density_kg_m3: 846.0, viscosity_
| 275 |             OilType::Engine0w20 => OilRuntimeProfile { label: "Engine 0W-20", density_kg_m3: 845.0, viscosity_in
| 276 |             OilType::Engine5w30 => OilRuntimeProfile { label: "Engine 5W-30", density_kg_m3: 855.0, viscosity_in
| 277 |             OilType::Engine5w40 => OilRuntimeProfile { label: "Engine 5W-40", density_kg_m3: 862.0, viscosity_in
| 278 |             OilType::Engine10w30 => OilRuntimeProfile { label: "Engine 10W-30", density_kg_m3: 870.0, viscosity_
| 279 |             OilType::Engine15w40 => OilRuntimeProfile { label: "Engine 15W-40", density_kg_m3: 880.0, viscosity_
| 280 |             OilType::Engine20w50 => OilRuntimeProfile { label: "Engine 20W-50", density_kg_m3: 892.0, viscosity_
| 281 |             OilType::TransformerInhibited => OilRuntimeProfile { label: "Transformer Oil Inhibited", density_kg_
| 282 |             OilType::Turbine32 => OilRuntimeProfile { label: "Turbine Oil 32", density_kg_m3: 855.0, viscosity_i
| 283 |             OilType::Turbine46 => OilRuntimeProfile { label: "Turbine Oil 46", density_kg_m3: 865.0, viscosity_i
| 284 |             OilType::Compressor46 => OilRuntimeProfile { label: "Compressor Oil 46", density_kg_m3: 862.0, visco
| 285 |             OilType::Compressor68 => OilRuntimeProfile { label: "Compressor Oil 68", density_kg_m3: 872.0, visco
| 286 |             OilType::BiodegradableHees46 => OilRuntimeProfile { label: "HEES Biodegradable 46", density_kg_m3: 9
| 287 |             OilType::BiodegradableHepr46 => OilRuntimeProfile { label: "HEPR Biodegradable 46", density_kg_m3: 9
| 288 |             OilType::FireResistantHfdu46 => OilRuntimeProfile { label: "HFDU Fire Resistant 46", density_kg_m3:
| 289 |             OilType::BrakeDot3 => OilRuntimeProfile { label: "Brake Fluid DOT3", density_kg_m3: 1035.0, viscosit
| 290 |             OilType::BrakeDot4 => OilRuntimeProfile { label: "Brake Fluid DOT4", density_kg_m3: 1060.0, viscosit
| 291 |             OilType::BrakeDot51 => OilRuntimeProfile { label: "Brake Fluid DOT5.1", density_kg_m3: 1045.0, visco
| 292 |             OilType::SyntheticEsters100 => OilRuntimeProfile { label: "Synthetic Ester 100", density_kg_m3: 915
| 293 |             OilType::RefrigerationMineral3gs => OilRuntimeProfile { label: "Refrigeration Mineral 3GS", density_
| 294 |             OilType::Pag46 => OilRuntimeProfile { label: "PAG 46", density_kg_m3: 995.0, viscosity_index: 130.0,
| 295 |             OilType::Pag100 => OilRuntimeProfile { label: "PAG 100", density_kg_m3: 1002.0, viscosity_index: 135
| 296 |             OilType::Pag150 => OilRuntimeProfile { label: "PAG 150", density_kg_m3: 1010.0, viscosity_index: 140
| 297 |             OilType::Poe32 => OilRuntimeProfile { label: "POE 32", density_kg_m3: 970.0, viscosity_index: 140.0,
| 298 |             OilType::Poe46 => OilRuntimeProfile { label: "POE 46", density_kg_m3: 975.0, viscosity_index: 143.0,
| 299 |             OilType::Poe68 => OilRuntimeProfile { label: "POE 68", density_kg_m3: 980.0, viscosity_index: 145.0,
| 300 |             OilType::Poe100 => OilRuntimeProfile { label: "POE 100", density_kg_m3: 988.0, viscosity_index: 148
| 301 |             OilType::VacuumPump100 => OilRuntimeProfile { label: "Vacuum Pump 100", density_kg_m3: 885.0, visco
| 302 |             OilType::Custom => OilRuntimeProfile { label: "Custom", density_kg_m3: 900.0, viscosity_index: 120.0
| 303 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 304 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 305 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 306 | (blank) | blank separator; no runtime behavior. |
| 307 | impl OilType { | implementation binding; behavior semantics and trait conformance live here. |
| 308 |     pub fn density_kg_m3(self) -> f64 { | function signature/body; changes affect API behavior and control-flow
| 309 |         self.runtime_profile().density_kg_m3 | support logic; impact depends on neighboring block and data flow
| 310 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 311 | (blank) | blank separator; no runtime behavior. |
| 312 |     pub fn viscosity_index(self) -> f64 { | function signature/body; changes affect API behavior and control-flo
| 313 |         self.runtime_profile().viscosity_index | support logic; impact depends on neighboring block and data flo
| 314 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 315 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 316 | (blank) | blank separator; no runtime behavior. |
| 317 | impl std::fmt::Display for LeakAlertLevel { | implementation binding; behavior semantics and trait conformance
| 318 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 319 |         let s = match self { | support logic; impact depends on neighboring block and data flow. |
| 320 |             LeakAlertLevel::Normal => "NORMAL", | support logic; impact depends on neighboring block and data fl
| 321 |             LeakAlertLevel::Watch => "WATCH", | support logic; impact depends on neighboring block and data flow
| 322 |             LeakAlertLevel::Warning => "WARNING", | support logic; impact depends on neighboring block and data
| 323 |             LeakAlertLevel::Critical => "CRITICAL", | support logic; impact depends on neighboring block and da
| 324 |             LeakAlertLevel::Ruptured => "RUPTURED", | support logic; impact depends on neighboring block and da
| 325 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 326 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 327 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 328 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 329 | (blank) | blank separator; no runtime behavior. |
| 330 | #[derive(Debug, Clone, Copy, PartialEq, Eq, Serialize, Deserialize)] | comment/documentation; changing affects
| 331 | pub enum OringMaterial { | state machine variants; changes ripple through matches and business logic. |
| 332 |     Nbr, | support logic; impact depends on neighboring block and data flow. |
```

```
| 333 |     Hnbr, | support logic; impact depends on neighboring block and data flow. |
| 334 |     Fkm, | support logic; impact depends on neighboring block and data flow. |
| 335 |     Epdm, | support logic; impact depends on neighboring block and data flow. |
| 336 |     Silicone, | support logic; impact depends on neighboring block and data flow. |
| 337 |     Fluorosilicone, | support logic; impact depends on neighboring block and data flow. |
| 338 |     Aflas, | support logic; impact depends on neighboring block and data flow. |
| 339 |     Ffkm, | support logic; impact depends on neighboring block and data flow. |
| 340 |     Ptfe, | support logic; impact depends on neighboring block and data flow. |
| 341 |     Polyurethane, | support logic; impact depends on neighboring block and data flow. |
| 342 |     Acm, | support logic; impact depends on neighboring block and data flow. |
| 343 |     Cr, | support logic; impact depends on neighboring block and data flow. |
| 344 |     Vmq, | support logic; impact depends on neighboring block and data flow. |
| 345 |     Fvmq, | support logic; impact depends on neighboring block and data flow. |
| 346 |     Sbr, | support logic; impact depends on neighboring block and data flow. |
| 347 |     Ecor, | support logic; impact depends on neighboring block and data flow. |
| 348 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 349 | (blank) | blank separator; no runtime behavior. |
| 350 | #[derive(Debug, Clone, Copy, Serialize, Deserialize)] | comment/documentation; changing affects understanding an
| 351 | pub struct MaterialRuntimeProfile { | data layout contract; field changes can break callers/serde/tests. |
| 352 |     pub label: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 353 |     pub min_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 354 |     pub max_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 355 |     pub gas_permeability_factor: f64, | support logic; impact depends on neighboring block and data flow. |
| 356 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 357 | (blank) | blank separator; no runtime behavior. |
| 358 | const ORING_MATERIAL_VARIANTS: [OringMaterial; 12] = [ | support logic; impact depends on neighboring block and
| 359 |     OringMaterial::Nbr, | support logic; impact depends on neighboring block and data flow. |
| 360 |     OringMaterial::Hnbr, | support logic; impact depends on neighboring block and data flow. |
| 361 |     OringMaterial::Fkm, | support logic; impact depends on neighboring block and data flow. |
| 362 |     OringMaterial::Epdm, | support logic; impact depends on neighboring block and data flow. |
| 363 |     OringMaterial::Silicone, | support logic; impact depends on neighboring block and data flow. |
| 364 |     OringMaterial::Fluorosilicone, | support logic; impact depends on neighboring block and data flow. |
| 365 |     OringMaterial::Aflas, | support logic; impact depends on neighboring block and data flow. |
| 366 |     OringMaterial::Ffkm, | support logic; impact depends on neighboring block and data flow. |
| 367 |     OringMaterial::Ptfe, | support logic; impact depends on neighboring block and data flow. |
| 368 |     OringMaterial::Polyurethane, | support logic; impact depends on neighboring block and data flow. |
| 369 |     OringMaterial::Acm, | support logic; impact depends on neighboring block and data flow. |
| 370 |     OringMaterial::Cr, | support logic; impact depends on neighboring block and data flow. |
| 371 | ]; | support logic; impact depends on neighboring block and data flow. |
| 372 | (blank) | blank separator; no runtime behavior. |
| 373 | impl OringMaterial { | implementation binding; behavior semantics and trait conformance live here. |
| 374 |     pub fn all() -> &'static [OringMaterial] { | function signature/body; changes affect API behavior and contro
| 375 |         &ORING_MATERIAL_VARIANTS | support logic; impact depends on neighboring block and data flow. |
| 376 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 377 | (blank) | blank separator; no runtime behavior. |
| 378 |     pub fn name(self) -> &'static str { | function signature/body; changes affect API behavior and control-flow
| 379 |         self.runtime_profile().label | support logic; impact depends on neighboring block and data flow. |
| 380 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 381 | (blank) | blank separator; no runtime behavior. |
| 382 |     pub fn runtime_profile(self) -> MaterialRuntimeProfile { | function signature/body; changes affect API behav
| 383 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 384 |             OringMaterial::Nbr => MaterialRuntimeProfile { label: "NBR", min_temp_c: -35.0, max_temp_c: 110.0, g
| 385 |             OringMaterial::Hnbr => MaterialRuntimeProfile { label: "HNBR", min_temp_c: -40.0, max_temp_c: 150.0,
| 386 |             OringMaterial::Fkm => MaterialRuntimeProfile { label: "FKM", min_temp_c: -20.0, max_temp_c: 200.0, g
| 387 |             OringMaterial::Epdm => MaterialRuntimeProfile { label: "EPDM", min_temp_c: -45.0, max_temp_c: 140.0,
| 388 |             OringMaterial::Silicone => MaterialRuntimeProfile { label: "VMQ Silicone", min_temp_c: -60.0, max_te
| 389 |             OringMaterial::Fluorosilicone => MaterialRuntimeProfile { label: "FVMQ Fluorosilicone", min_temp_c:
| 390 |             OringMaterial::Aflas => MaterialRuntimeProfile { label: "AFLAS", min_temp_c: -10.0, max_temp_c: 230
| 391 |             OringMaterial::Ffkm => MaterialRuntimeProfile { label: "FFKM", min_temp_c: -15.0, max_temp_c: 320.0,
| 392 |             OringMaterial::Ptfe => MaterialRuntimeProfile { label: "PTFE", min_temp_c: -80.0, max_temp_c: 260.0,
| 393 |             OringMaterial::Polyurethane => MaterialRuntimeProfile { label: "PU", min_temp_c: -35.0, max_temp_c:
| 394 |             OringMaterial::Acm => MaterialRuntimeProfile { label: "ACM", min_temp_c: -25.0, max_temp_c: 165.0, g
| 395 |             OringMaterial::Cr => MaterialRuntimeProfile { label: "CR", min_temp_c: -40.0, max_temp_c: 115.0, gas
| 396 |             OringMaterial::Vmq => MaterialRuntimeProfile { label: "VMQ", min_temp_c: -60.0, max_temp_c: 205.0, g
| 397 |             OringMaterial::Fvmq => MaterialRuntimeProfile { label: "FVMQ", min_temp_c: -58.0, max_temp_c: 190.0,
| 398 |             OringMaterial::Sbr => MaterialRuntimeProfile { label: "SBR", min_temp_c: -40.0, max_temp_c: 100.0, g
| 399 |             OringMaterial::Ecor => MaterialRuntimeProfile { label: "ECO", min_temp_c: -35.0, max_temp_c: 140.0,
| 400 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 401 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 402 | (blank) | blank separator; no runtime behavior. |
| 403 |     pub fn temp_window_c(self) -> (f64, f64) { | function signature/body; changes affect API behavior and contro
| 404 |         let p = self.runtime_profile(); | support logic; impact depends on neighboring block and data flow. |
| 405 |         (p.min_temp_c, p.max_temp_c) | support logic; impact depends on neighboring block and data flow. |
| 406 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 407 | (blank) | blank separator; no runtime behavior. |
| 408 |     pub fn gas_permeability_factor(self) -> f64 { | function signature/body; changes affect API behavior and co
```

```
| 409 |         self.runtime_profile().gas_permeability_factor | support logic; impact depends on neighboring block and
| 410 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 411 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 412 | (blank) | blank separator; no runtime behavior. |
| 413 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 414 | pub struct MaterialCatalogEntry { | data layout contract; field changes can break callers/serde/tests. |
| 415 |     pub id: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 416 |     pub family: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 417 |     pub designation: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 418 |     pub hardness_shore: f64, | support logic; impact depends on neighboring block and data flow. |
| 419 |     pub tensile_strength_mpa: f64, | support logic; impact depends on neighboring block and data flow. |
| 420 |     pub elongation_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 421 |     pub compression_set_pct_70h: f64, | support logic; impact depends on neighboring block and data flow. |
| 422 |     pub tear_strength_kn_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 423 |     pub density_g_cm3: f64, | support logic; impact depends on neighboring block and data flow. |
| 424 |     pub temp_min_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 425 |     pub temp_max_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 426 |     pub abrasion_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 427 |     pub corrosion_resistance_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 428 |     pub hydraulic_oil_compat: f64, | support logic; impact depends on neighboring block and data flow. |
| 429 |     pub engine_oil_compat: f64, | support logic; impact depends on neighboring block and data flow. |
| 430 |     pub refrigerant_oil_compat: f64, | support logic; impact depends on neighboring block and data flow. |
| 431 |     pub notes: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 432 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 433 | (blank) | blank separator; no runtime behavior. |
| 434 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 435 | pub struct OilCatalogEntry { | data layout contract; field changes can break callers/serde/tests. |
| 436 |     pub id: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 437 |     pub family: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 438 |     pub grade: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 439 |     pub base_stock: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 440 |     pub density_15c_kg_m3: f64, | support logic; impact depends on neighboring block and data flow. |
| 441 |     pub viscosity_40c_cst: f64, | support logic; impact depends on neighboring block and data flow. |
| 442 |     pub viscosity_100c_cst: f64, | support logic; impact depends on neighboring block and data flow. |
| 443 |     pub viscosity_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 444 |     pub pour_point_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 445 |     pub flash_point_c: f64, | protocol or I/O critical; changes may impact external integration correctness. |
| 446 |     pub tan_mg_koh_g: f64, | support logic; impact depends on neighboring block and data flow. |
| 447 |     pub water_content_ppm_limit: f64, | support logic; impact depends on neighboring block and data flow. |
| 448 |     pub demulsibility_min: f64, | support logic; impact depends on neighboring block and data flow. |
| 449 |     pub copper_corrosion_level: f64, | support logic; impact depends on neighboring block and data flow. |
| 450 |     pub oxidation_stability_h: f64, | support logic; impact depends on neighboring block and data flow. |
| 451 |     pub recommended_min_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 452 |     pub recommended_max_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 453 |     pub anti_wear_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 454 |     pub corrosion_inhibition_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 455 |     pub notes: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 456 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 457 | (blank) | blank separator; no runtime behavior. |
| 458 | pub fn engineering_material_catalog() -> Vec<MaterialCatalogEntry> { | function signature/body; changes affect A
| 459 |     vec![ | support logic; impact depends on neighboring block and data flow. |
| 460 |         MaterialCatalogEntry { id: "OR-NBR-70", family: "O-RING", designation: "NBR 70", hardness_shore: 70.0, t
| 461 |         MaterialCatalogEntry { id: "OR-NBR-90", family: "O-RING", designation: "NBR 90", hardness_shore: 90.0, t
| 462 |         MaterialCatalogEntry { id: "OR-HNBR-80", family: "O-RING", designation: "HNBR 80", hardness_shore: 80.0,
| 463 |         MaterialCatalogEntry { id: "OR-FKM-75", family: "O-RING", designation: "FKM 75", hardness_shore: 75.0, t
| 464 |         MaterialCatalogEntry { id: "OR-EPDM-70", family: "O-RING", designation: "EPDM 70", hardness_shore: 70.0,
| 465 |         MaterialCatalogEntry { id: "OR-AFLAS-80", family: "O-RING", designation: "AFLAS 80", hardness_shore: 80
| 466 |         MaterialCatalogEntry { id: "OR-FFKM-75", family: "O-RING", designation: "FFKM 75", hardness_shore: 75.0,
| 467 |         MaterialCatalogEntry { id: "OR-SIL-70", family: "O-RING", designation: "VMQ 70", hardness_shore: 70.0, t
| 468 |         MaterialCatalogEntry { id: "OR-FSIL-70", family: "O-RING", designation: "FVMQ 70", hardness_shore: 70.0,
| 469 |         MaterialCatalogEntry { id: "OR-PTFE", family: "O-RING", designation: "PTFE Virgin", hardness_shore: 60.0
| 470 |         MaterialCatalogEntry { id: "OR-PU-93", family: "O-RING", designation: "PU 93A", hardness_shore: 93.0, te
| 471 |         MaterialCatalogEntry { id: "OR-ACM-70", family: "O-RING", designation: "ACM 70", hardness_shore: 70.0, t
| 472 |         MaterialCatalogEntry { id: "HS-NBR-TUBE", family: "HOSE_INNER_TUBE", designation: "NBR Tube", hardness_s
| 473 |         MaterialCatalogEntry { id: "HS-HNBR-TUBE", family: "HOSE_INNER_TUBE", designation: "HNBR Tube", hardness
| 474 |         MaterialCatalogEntry { id: "HS-CSM-COVER", family: "HOSE_COVER", designation: "CSM Cover", hardness_shor
| 475 |         MaterialCatalogEntry { id: "HS-PTFE-TUBE", family: "HOSE_INNER_TUBE", designation: "PTFE Hose Tube", har
| 476 |         MaterialCatalogEntry { id: "HS-ARAMID-BRAID", family: "HOSE_REINFORCEMENT", designation: "Aramid Braid",
| 477 |         MaterialCatalogEntry { id: "HS-STEEL-WIRE", family: "HOSE_REINFORCEMENT", designation: "High-tensile Ste
| 478 |         MaterialCatalogEntry { id: "SL-PTFE-LIP", family: "DYNAMIC_SEAL", designation: "PTFE Lip Seal", hardness
| 479 |         MaterialCatalogEntry { id: "SL-PU-LIP", family: "DYNAMIC_SEAL", designation: "PU Lip Seal", hardness_sho
| 480 |         MaterialCatalogEntry { id: "SL-NBR-LIP", family: "DYNAMIC_SEAL", designation: "NBR Lip", hardness_shore
| 481 |         MaterialCatalogEntry { id: "BK-PTFE-GLASS", family: "BACKUP_RING", designation: "PTFE 25% Glass", hardne
| 482 |         MaterialCatalogEntry { id: "BK-PA66", family: "BACKUP_RING", designation: "PA66", hardness_shore: 80.0,
| 483 |         MaterialCatalogEntry { id: "BK-PEEK", family: "BACKUP_RING", designation: "PEEK", hardness_shore: 85.0,
| 484 |         MaterialCatalogEntry { id: "MS-CARBON", family: "MECH_SEAL_FACE", designation: "Carbon Graphite", hardne
```

```
| 485 |         MaterialCatalogEntry { id: "MS-SIC", family: "MECH_SEAL_FACE", designation: "Silicon Carbide", hardness_
| 486 |         MaterialCatalogEntry { id: "MS-TC", family: "MECH_SEAL_FACE", designation: "Tungsten Carbide", hardness_
| 487 |         MaterialCatalogEntry { id: "WR-POM", family: "WEAR_RING", designation: "POM Wear Ring", hardness_shore:
| 488 |         MaterialCatalogEntry { id: "WR-PTFE-BRONZE", family: "WEAR_RING", designation: "PTFE Bronze Filled", har
| 489 |         MaterialCatalogEntry { id: "WR-PHENOLIC", family: "WEAR_RING", designation: "Fabric Phenolic", hardness_
| 490 |         MaterialCatalogEntry { id: "OR-ECO-80", family: "O-RING", designation: "ECO 80", hardness_shore: 80.0, t
| 491 |         MaterialCatalogEntry { id: "OR-CR-70", family: "O-RING", designation: "CR 70", hardness_shore: 70.0, ter
| 492 |         MaterialCatalogEntry { id: "OR-VMQ-50", family: "O-RING", designation: "VMQ 50", hardness_shore: 50.0, t
| 493 |     ] | support logic; impact depends on neighboring block and data flow. |
| 494 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 495 | (blank) | blank separator; no runtime behavior. |
| 496 | pub fn engineering_oil_catalog() -> Vec<OilCatalogEntry> { | function signature/body; changes affect API behavio
| 497 |     vec![ | support logic; impact depends on neighboring block and data flow. |
| 498 |         OilCatalogEntry { id: "HYD-ISO-22", family: "HYDRAULIC", grade: "ISO VG 22", base_stock: "Mineral", dens
| 499 |         OilCatalogEntry { id: "HYD-ISO-32", family: "HYDRAULIC", grade: "ISO VG 32", base_stock: "Mineral", dens
| 500 |         OilCatalogEntry { id: "HYD-ISO-46", family: "HYDRAULIC", grade: "ISO VG 46", base_stock: "Mineral", dens
| 501 |         OilCatalogEntry { id: "HYD-ISO-68", family: "HYDRAULIC", grade: "ISO VG 68", base_stock: "Mineral", dens
| 502 |         OilCatalogEntry { id: "HYD-ISO-100", family: "HYDRAULIC", grade: "ISO VG 100", base_stock: "Mineral", de
| 503 |         OilCatalogEntry { id: "AW-46", family: "HYDRAULIC_AW", grade: "AW 46", base_stock: "Mineral", density_15
| 504 |         OilCatalogEntry { id: "HEES-46", family: "BIODEGRADABLE", grade: "HEES 46", base_stock: "Synthetic Este
| 505 |         OilCatalogEntry { id: "HEPR-46", family: "BIODEGRADABLE", grade: "HEPR 46", base_stock: "PAO", density_1
| 506 |         OilCatalogEntry { id: "HFDU-46", family: "FIRE_RESISTANT", grade: "HFDU 46", base_stock: "Synthetic Este
| 507 |         OilCatalogEntry { id: "ENG-0W20", family: "ENGINE", grade: "0W-20", base_stock: "Group III/PAO", densit
| 508 |         OilCatalogEntry { id: "ENG-5W30", family: "ENGINE", grade: "5W-30", base_stock: "Group II/III", density_
| 509 |         OilCatalogEntry { id: "ENG-10W30", family: "ENGINE", grade: "10W-30", base_stock: "Group II", density_15
| 510 |         OilCatalogEntry { id: "ENG-15W40", family: "ENGINE", grade: "15W-40", base_stock: "Group II", density_15
| 511 |         OilCatalogEntry { id: "ENG-20W50", family: "ENGINE", grade: "20W-50", base_stock: "Group II", density_15
| 512 |         OilCatalogEntry { id: "ATF-DIII", family: "TRANSMISSION", grade: "Dexron III", base_stock: "Mineral/Syn
| 513 |         OilCatalogEntry { id: "ATF-DVI", family: "TRANSMISSION", grade: "Dexron VI", base_stock: "Synthetic", de
| 514 |         OilCatalogEntry { id: "GEAR-75W90", family: "GEAR", grade: "75W-90", base_stock: "PAO", density_15c_kg_m
| 515 |         OilCatalogEntry { id: "GEAR-80W90", family: "GEAR", grade: "80W-90", base_stock: "Mineral", density_15c_
| 516 |         OilCatalogEntry { id: "PAG-46", family: "REFRIGERATION", grade: "PAG 46", base_stock: "Polyalkylene Glyc
| 517 |         OilCatalogEntry { id: "PAG-100", family: "REFRIGERATION", grade: "PAG 100", base_stock: "Polyalkylene Gl
| 518 |         OilCatalogEntry { id: "PAG-150", family: "REFRIGERATION", grade: "PAG 150", base_stock: "Polyalkylene Gl
| 519 |         OilCatalogEntry { id: "POE-32", family: "REFRIGERATION", grade: "POE 32", base_stock: "Polyol Ester", de
| 520 |         OilCatalogEntry { id: "POE-46", family: "REFRIGERATION", grade: "POE 46", base_stock: "Polyol Ester", de
| 521 |         OilCatalogEntry { id: "POE-68", family: "REFRIGERATION", grade: "POE 68", base_stock: "Polyol Ester", de
| 522 |         OilCatalogEntry { id: "POE-100", family: "REFRIGERATION", grade: "POE 100", base_stock: "Polyol Ester",
| 523 |         OilCatalogEntry { id: "TRF-INH", family: "ELECTRICAL", grade: "Transformer Inhibited", base_stock: "Mine
| 524 |         OilCatalogEntry { id: "TURB-32", family: "TURBINE", grade: "ISO 32", base_stock: "Mineral", density_15c_
| 525 |         OilCatalogEntry { id: "TURB-46", family: "TURBINE", grade: "ISO 46", base_stock: "Mineral", density_15c_
| 526 |         OilCatalogEntry { id: "COMP-46", family: "COMPRESSOR", grade: "ISO 46", base_stock: "Mineral", density_1
| 527 |         OilCatalogEntry { id: "COMP-68", family: "COMPRESSOR", grade: "ISO 68", base_stock: "Mineral", density_1
| 528 |         OilCatalogEntry { id: "DOT3", family: "BRAKE", grade: "DOT 3", base_stock: "Glycol Ether", density_15c_k
| 529 |         OilCatalogEntry { id: "DOT4", family: "BRAKE", grade: "DOT 4", base_stock: "Borate Ester", density_15c_k
| 530 |         OilCatalogEntry { id: "DOT5.1", family: "BRAKE", grade: "DOT 5.1", base_stock: "Borate Ester", density_1
| 531 |         OilCatalogEntry { id: "UTTO-10W30", family: "TRACTOR", grade: "UTTO 10W-30", base_stock: "Mineral", dens
| 532 |         OilCatalogEntry { id: "UTTO-80W", family: "TRACTOR", grade: "UTTO 80W", base_stock: "Mineral", density_1
| 533 |         OilCatalogEntry { id: "VAC-100", family: "VACUUM", grade: "ISO 100", base_stock: "Mineral", density_15c_
| 534 |         OilCatalogEntry { id: "ESTER-100", family: "SYNTHETIC", grade: "ISO 100", base_stock: "Synthetic Ester",
| 535 |         OilCatalogEntry { id: "REF-MIN-3GS", family: "REFRIGERATION", grade: "3GS", base_stock: "Mineral Naphthe
| 536 |     ] | support logic; impact depends on neighboring block and data flow. |
| 537 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 538 | (blank) | blank separator; no runtime behavior. |
| 539 | #[derive(Debug, Clone, Copy, Serialize, Deserialize)] | comment/documentation; changing affects understanding an
| 540 | pub struct PressureEnvelope { | data layout contract; field changes can break callers/serde/tests. |
| 541 |     pub min_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 542 |     pub mean_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 543 |     pub ideal_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 544 |     pub max_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 545 |     pub rupture_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 546 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 547 | (blank) | blank separator; no runtime behavior. |
| 548 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 549 | pub struct OringSpec { | data layout contract; field changes can break callers/serde/tests. |
| 550 |     pub id_tag: String, | support logic; impact depends on neighboring block and data flow. |
| 551 |     pub material: OringMaterial, | support logic; impact depends on neighboring block and data flow. |
| 552 |     pub shore_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 553 |     pub cross_section_mm: f64, | support logic; impact depends on neighboring block and data flow. |
| 554 |     pub squeeze_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 555 |     pub extrusion_gap_mm: f64, | support logic; impact depends on neighboring block and data flow. |
| 556 |     pub compression_set_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 557 |     pub design_life_hours: f64, | support logic; impact depends on neighboring block and data flow. |
| 558 |     pub base_leak_area_mm2: f64, | support logic; impact depends on neighboring block and data flow. |
| 559 |     pub discharge_coeff: f64, | support logic; impact depends on neighboring block and data flow. |
| 560 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 561 | (blank) | blank separator; no runtime behavior. |
| 562 | #[derive(Debug, Clone, Copy, Serialize, Deserialize)] | comment/documentation; changing affects understanding an
| 563 | pub struct CircuitInput { | data layout contract; field changes can break callers/serde/tests. |
| 564 |     pub pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 565 |     pub delta_p_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 566 |     pub temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 567 |     pub cycles_per_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 568 |     pub duty_01: f64, | support logic; impact depends on neighboring block and data flow. |
| 569 |     pub fluid_density_kg_m3: f64, | support logic; impact depends on neighboring block and data flow. |
| 570 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 571 | (blank) | blank separator; no runtime behavior. |
| 572 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 573 | pub struct CircuitResult { | data layout contract; field changes can break callers/serde/tests. |
| 574 |     pub name: String, | support logic; impact depends on neighboring block and data flow. |
| 575 |     pub alert: LeakAlertLevel, | support logic; impact depends on neighboring block and data flow. |
| 576 |     pub pressure_band: PressureBand, | support logic; impact depends on neighboring block and data flow. |
| 577 |     pub current_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 578 |     pub rupture_probability_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 579 |     pub predicted_hours_to_rupture: Option<f64>, | support logic; impact depends on neighboring block and data f
| 580 |     pub rupture_pressure_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow.
| 581 |     pub rupture_elapsed_h: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 582 |     pub leak_area_mm2: f64, | support logic; impact depends on neighboring block and data flow. |
| 583 |     pub leak_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 584 |     pub pressure_min_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 585 |     pub pressure_mean_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 586 |     pub pressure_max_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 587 |     pub recommended_hold_min_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 588 |     pub recommended_hold_target_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 589 |     pub recommended_hold_max_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 590 |     pub margin_to_max_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 591 |     pub margin_to_rupture_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 592 |     pub rca_hint: String, | support logic; impact depends on neighboring block and data flow. |
| 593 |     pub pca_recommendation: String, | support logic; impact depends on neighboring block and data flow. |
| 594 |     pub rupture_confirmed: bool, | support logic; impact depends on neighboring block and data flow. |
| 595 |     pub warning_text: String, | support logic; impact depends on neighboring block and data flow. |
| 596 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 597 | (blank) | blank separator; no runtime behavior. |
| 598 | #[derive(Debug, Clone, Copy, Serialize, Deserialize)] | comment/documentation; changing affects understanding an
| 599 | pub struct ManualCircuitParams { | data layout contract; field changes can break callers/serde/tests. |
| 600 |     pub oil_type: Option<OilType>, | support logic; impact depends on neighboring block and data flow. |
| 601 |     pub piston_pressure_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 602 |     pub operation_pressure_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow
| 603 |     pub pressure_min_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 604 |     pub pressure_mean_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 605 |     pub pressure_ideal_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 606 |     pub pressure_max_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 607 |     pub pressure_rupture_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow.
| 608 |     pub oring_squeeze_pct: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 609 |     pub compression_set_pct: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 610 |     pub base_leak_area_mm2: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 611 |     pub max_supported_temp_c: Option<f64>, | support logic; impact depends on neighboring block and data flow.
| 612 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 613 | (blank) | blank separator; no runtime behavior. |
| 614 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 615 | pub struct ScenarioPrediction { | data layout contract; field changes can break callers/serde/tests. |
| 616 |     pub circuit_name: String, | support logic; impact depends on neighboring block and data flow. |
| 617 |     pub scenario_name: String, | support logic; impact depends on neighboring block and data flow. |
| 618 |     pub peak_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 619 |     pub final_alert: LeakAlertLevel, | support logic; impact depends on neighboring block and data flow. |
| 620 |     pub final_rupture_probability_pct: f64, | support logic; impact depends on neighboring block and data flow.
| 621 |     pub hours_to_rupture: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 622 |     pub likely_failure_mode: String, | support logic; impact depends on neighboring block and data flow. |
| 623 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 624 | (blank) | blank separator; no runtime behavior. |
| 625 | #[derive(Debug, Clone, Serialize, Deserialize)] | comment/documentation; changing affects understanding and inte
| 626 | pub struct LeakAlert { | data layout contract; field changes can break callers/serde/tests. |
| 627 |     pub circuit_name: String, | support logic; impact depends on neighboring block and data flow. |
| 628 |     pub level: LeakAlertLevel, | support logic; impact depends on neighboring block and data flow. |
| 629 |     pub message: String, | support logic; impact depends on neighboring block and data flow. |
| 630 |     pub predicted_hours_to_rupture: Option<f64>, | support logic; impact depends on neighboring block and data f
| 631 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 632 | (blank) | blank separator; no runtime behavior. |
| 633 | #[derive(Debug, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceability.
| 634 | struct PressureStats { | data layout contract; field changes can break callers/serde/tests. |
| 635 |     min_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 636 |     max_bar: f64, | support logic; impact depends on neighboring block and data flow. |
```

```
| 637 |     mean_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 638 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 639 | (blank) | blank separator; no runtime behavior. |
| 640 | impl PressureStats { | implementation binding; behavior semantics and trait conformance live here. |
| 641 |     fn new(seed_bar: f64) -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 642 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 643 |             min_bar: seed_bar, | support logic; impact depends on neighboring block and data flow. |
| 644 |             max_bar: seed_bar, | support logic; impact depends on neighboring block and data flow. |
| 645 |             mean_bar: seed_bar, | support logic; impact depends on neighboring block and data flow. |
| 646 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 647 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 648 | (blank) | blank separator; no runtime behavior. |
| 649 |     fn update(&mut self, p_bar: f64) { | function signature/body; changes affect API behavior and control-flow.
| 650 |         self.min_bar = self.min_bar.min(p_bar); | support logic; impact depends on neighboring block and data f
| 651 |         self.max_bar = self.max_bar.max(p_bar); | support logic; impact depends on neighboring block and data f
| 652 |         self.mean_bar = self.mean_bar * 0.992 + p_bar * 0.008; | support logic; impact depends on neighboring b
| 653 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 654 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 655 | (blank) | blank separator; no runtime behavior. |
| 656 | #[derive(Debug, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceability.
| 657 | struct SealState { | data layout contract; field changes can break callers/serde/tests. |
| 658 |     damage_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 659 |     extrusion_index: f64, | support logic; impact depends on neighboring block and data flow. |
| 660 |     thermal_damage: f64, | support logic; impact depends on neighboring block and data flow. |
| 661 |     cumulative_overpressure_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 662 |     total_operating_h: f64, | support logic; impact depends on neighboring block and data flow. |
| 663 |     leak_area_mm2: f64, | support logic; impact depends on neighboring block and data flow. |
| 664 |     leak_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 665 |     rupture_confirmed: bool, | support logic; impact depends on neighboring block and data flow. |
| 666 |     rupture_pressure_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 667 |     rupture_elapsed_h: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 668 |     rupture_probability_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 669 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 670 | (blank) | blank separator; no runtime behavior. |
| 671 | impl SealState { | implementation binding; behavior semantics and trait conformance live here. |
| 672 |     fn new(base_area_mm2: f64) -> Self { | function signature/body; changes affect API behavior and control-flo
| 673 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 674 |             damage_index: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 675 |             extrusion_index: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 676 |             thermal_damage: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 677 |             cumulative_overpressure_s: 0.0, | support logic; impact depends on neighboring block and data flow.
| 678 |             total_operating_h: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 679 |             leak_area_mm2: base_area_mm2, | support logic; impact depends on neighboring block and data flow. |
| 680 |             leak_lpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 681 |             rupture_confirmed: false, | support logic; impact depends on neighboring block and data flow. |
| 682 |             rupture_pressure_bar: None, | support logic; impact depends on neighboring block and data flow. |
| 683 |             rupture_elapsed_h: None, | support logic; impact depends on neighboring block and data flow. |
| 684 |             rupture_probability_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 685 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 686 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 687 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 688 | (blank) | blank separator; no runtime behavior. |
| 689 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 690 | pub struct LeakCircuit { | data layout contract; field changes can break callers/serde/tests. |
| 691 |     pub name: String, | support logic; impact depends on neighboring block and data flow. |
| 692 |     pub application: String, | support logic; impact depends on neighboring block and data flow. |
| 693 |     pub component: CircuitComponent, | support logic; impact depends on neighboring block and data flow. |
| 694 |     pub oil_type: OilType, | support logic; impact depends on neighboring block and data flow. |
| 695 |     pub seal_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 696 |     pub piston_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 697 |     pub operation_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 698 |     pub pressure: PressureEnvelope, | support logic; impact depends on neighboring block and data flow. |
| 699 |     pub spec: OringSpec, | support logic; impact depends on neighboring block and data flow. |
| 700 |     pub coeffs: LeakModelCoefficients, | support logic; impact depends on neighboring block and data flow. |
| 701 |     pub reservoir_volume_l: f64, | support logic; impact depends on neighboring block and data flow. |
| 702 |     pub pressure_support_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 703 |     stats: PressureStats, | support logic; impact depends on neighboring block and data flow. |
| 704 |     state: SealState, | support logic; impact depends on neighboring block and data flow. |
| 705 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 706 | (blank) | blank separator; no runtime behavior. |
| 707 | impl LeakCircuit { | implementation binding; behavior semantics and trait conformance live here. |
| 708 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent tra
| 709 |     pub fn new( | function signature/body; changes affect API behavior and control-flow. |
| 710 |         name: &str, | support logic; impact depends on neighboring block and data flow. |
| 711 |         application: &str, | support logic; impact depends on neighboring block and data flow. |
| 712 |         component: CircuitComponent, | support logic; impact depends on neighboring block and data flow. |
```

```
| 713 |         oil_type: OilType, | support logic; impact depends on neighboring block and data flow. |
| 714 |         seal_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 715 |         piston_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 716 |         operation_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 717 |         pressure: PressureEnvelope, | support logic; impact depends on neighboring block and data flow. |
| 718 |         spec: OringSpec, | support logic; impact depends on neighboring block and data flow. |
| 719 |         reservoir_volume_l: f64, | support logic; impact depends on neighboring block and data flow. |
| 720 |         pressure_support_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 721 |     ) -> Self { | support logic; impact depends on neighboring block and data flow. |
| 722 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 723 |             name: name.to_string(), | support logic; impact depends on neighboring block and data flow. |
| 724 |             application: application.to_string(), | support logic; impact depends on neighboring block and data
| 725 |             component, | support logic; impact depends on neighboring block and data flow. |
| 726 |             oil_type, | support logic; impact depends on neighboring block and data flow. |
| 727 |             seal_count, | support logic; impact depends on neighboring block and data flow. |
| 728 |             piston_pressure_bar, | support logic; impact depends on neighboring block and data flow. |
| 729 |             operation_pressure_bar, | support logic; impact depends on neighboring block and data flow. |
| 730 |             pressure, | support logic; impact depends on neighboring block and data flow. |
| 731 |             coeffs: LeakModelCoefficients::default(), | support logic; impact depends on neighboring block and
| 732 |             stats: PressureStats::new(pressure.mean_bar), | support logic; impact depends on neighboring block
| 733 |             state: SealState::new(spec.base_leak_area_mm2), | support logic; impact depends on neighboring bloc
| 734 |             spec, | support logic; impact depends on neighboring block and data flow. |
| 735 |             reservoir_volume_l, | support logic; impact depends on neighboring block and data flow. |
| 736 |             pressure_support_lpm, | support logic; impact depends on neighboring block and data flow. |
| 737 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 738 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 739 | (blank) | blank separator; no runtime behavior. |
| 740 |     pub fn step(&mut self, input: CircuitInput, dt: f64) -> CircuitResult { | function signature/body; changes
| 741 |         let dt = dt.max(1e-6); | support logic; impact depends on neighboring block and data flow. |
| 742 |         self.stats.update(input.pressure_bar.max(0.0)); | support logic; impact depends on neighboring block an
| 743 |         self.state.total_operating_h += dt / 3600.0; | support logic; impact depends on neighboring block and d
| 744 | (blank) | blank separator; no runtime behavior. |
| 745 |         let (t_min, t_max) = self.spec.material.temp_window_c(); | support logic; impact depends on neighboring
| 746 |         let temp_excess = if input.temp_c > t_max { | support logic; impact depends on neighboring block and da
| 747 |             (input.temp_c - t_max) / 20.0 | support logic; impact depends on neighboring block and data flow. |
| 748 |         } else if input.temp_c < t_min { | support logic; impact depends on neighboring block and data flow. |
| 749 |             (t_min - input.temp_c) / 25.0 | support logic; impact depends on neighboring block and data flow. |
| 750 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 751 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 752 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 753 | (blank) | blank separator; no runtime behavior. |
| 754 |         let pressure_ratio_max = input.pressure_bar / self.pressure.max_bar.max(1e-3); | support logic; impact
| 755 |         let pressure_ratio_rupt = input.pressure_bar / self.pressure.rupture_bar.max(1e-3); | support logic; im
| 756 | (blank) | blank separator; no runtime behavior. |
| 757 |         if input.pressure_bar > self.pressure.max_bar { | runtime gate; condition changes can enable/disable cr
| 758 |             self.state.cumulative_overpressure_s += dt; | support logic; impact depends on neighboring block an
| 759 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 760 | (blank) | blank separator; no runtime behavior. |
| 761 |         let squeeze_target = 0.20; | support logic; impact depends on neighboring block and data flow. |
| 762 |         let squeeze = (self.spec.squeeze_pct / 100.0).clamp(0.05, 0.45); | support logic; impact depends on nei
| 763 |         let squeeze_penalty = ((squeeze - squeeze_target).abs() / squeeze_target).clamp(0.0, 2.0); | support lo
| 764 | (blank) | blank separator; no runtime behavior. |
| 765 |         let hardness_penalty = ((80.0 - self.spec.shore_a).max(0.0) / 40.0).clamp(0.0, 1.0); | support logic; i
| 766 |         let gap_penalty = (self.spec.extrusion_gap_mm / 0.25).clamp(0.2, 3.0); | support logic; impact depends
| 767 |         let age_ratio = | support logic; impact depends on neighboring block and data flow. |
| 768 |             (self.state.total_operating_h / self.spec.design_life_hours.max(1.0)).clamp(0.0, 4.0); | support lo
| 769 |         let cycle_factor = (input.cycles_per_s / 2.0).clamp(0.0, 4.0); | support logic; impact depends on neigh
| 770 |         let duty = input.duty_01.clamp(0.0, 1.0); | support logic; impact depends on neighboring block and data
| 771 |         let oil_vi = self.oil_type.viscosity_index(); | support logic; impact depends on neighboring block and
| 772 |         let vi_factor = (130.0 / oil_vi.max(60.0)).clamp(0.7, 1.8); | support logic; impact depends on neighbor
| 773 |         let piston_overload = | support logic; impact depends on neighboring block and data flow. |
| 774 |             (self.piston_pressure_bar / self.pressure.ideal_bar.max(1e-3)).clamp(0.4, 2.2); | support logic; im
| 775 |         let operation_stress = | support logic; impact depends on neighboring block and data flow. |
| 776 |             (self.operation_pressure_bar / self.pressure.ideal_bar.max(1e-3)).clamp(0.4, 2.5); | support logic;
| 777 | (blank) | blank separator; no runtime behavior. |
| 778 |         let mut damage_rate = 2.2e-6 * self.coeffs.damage_rate_scale.max(0.05); | support logic; impact depends
| 779 |         damage_rate *= 1.0 + 1.9 * (pressure_ratio_max - 0.85).max(0.0).powi(2); | support logic; impact depend
| 780 |         damage_rate *= 1.0 + 0.75 * temp_excess.max(0.0); | support logic; impact depends on neighboring block
| 781 |         damage_rate *= 1.0 + 0.45 * squeeze_penalty; | support logic; impact depends on neighboring block and d
| 782 |         damage_rate *= 1.0 + 0.35 * cycle_factor; | support logic; impact depends on neighboring block and data
| 783 |         damage_rate *= 1.0 + 0.28 * age_ratio; | support logic; impact depends on neighboring block and data fl
| 784 |         damage_rate *= 0.8 + 0.6 * duty; | support logic; impact depends on neighboring block and data flow. |
| 785 |         damage_rate *= vi_factor; | support logic; impact depends on neighboring block and data flow. |
| 786 |         damage_rate *= 0.75 + 0.25 * piston_overload; | support logic; impact depends on neighboring block and
| 787 |         damage_rate *= 0.75 + 0.25 * operation_stress; | support logic; impact depends on neighboring block and
| 788 | (blank) | blank separator; no runtime behavior. |
```

```
| 789 |         self.state.damage_index = (self.state.damage_index + damage_rate * dt).clamp(0.0, 1.6); | support logic
| 790 | (blank) | blank separator; no runtime behavior. |
| 791 |         let extrusion_drive = | support logic; impact depends on neighboring block and data flow. |
| 792 |             (pressure_ratio_max - 0.9).max(0.0) * (1.0 + hardness_penalty) * gap_penalty; | support logic; impac
| 793 |         self.state.extrusion_index = (self.state.extrusion_index | support logic; impact depends on neighboring
| 794 |             + extrusion_drive * dt * 1.8e-4 * self.coeffs.extrusion_rate_scale.max(0.05)) | support logic; impac
| 795 |             .clamp(0.0, 1.5); | support logic; impact depends on neighboring block and data flow. |
| 796 | (blank) | blank separator; no runtime behavior. |
| 797 |         let thermal_rate = (temp_excess.max(0.0) * (0.9 + 0.4 * duty)) | support logic; impact depends on neigh
| 798 |             * 2.5e-5 | support logic; impact depends on neighboring block and data flow. |
| 799 |             * self.coeffs.thermal_rate_scale.max(0.05); | support logic; impact depends on neighboring block an
| 800 |         self.state.thermal_damage = (self.state.thermal_damage + thermal_rate * dt).clamp(0.0, 1.2); | support
| 801 | (blank) | blank separator; no runtime behavior. |
| 802 |         if matches!(self.component, CircuitComponent::AcHose) { | runtime gate; condition changes can enable/di
| 803 |             self.state.extrusion_index = (self.state.extrusion_index | support logic; impact depends on neighbo
| 804 |                 + (pressure_ratio_max - 0.75).max(0.0) | support logic; impact depends on neighboring block and
| 805 |                     * dt | support logic; impact depends on neighboring block and data flow. |
| 806 |                     * 2.2e-4 | support logic; impact depends on neighboring block and data flow. |
| 807 |                     * self.coeffs.extrusion_rate_scale.max(0.05)) | support logic; impact depends on neighborin
| 808 |                 .clamp(0.0, 1.8); | support logic; impact depends on neighboring block and data flow. |
| 809 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 810 | (blank) | blank separator; no runtime behavior. |
| 811 |         if !self.state.rupture_confirmed { | runtime gate; condition changes can enable/disable critical paths.
| 812 |             let immediate_burst = input.pressure_bar >= self.pressure.rupture_bar; | support logic; impact depe
| 813 |             let fatigue_burst = self.state.damage_index > 1.0 && self.state.extrusion_index > 0.8; | support lo
| 814 |             let thermal_burst = self.state.thermal_damage > 1.0 && pressure_ratio_max > 0.95; | support logic;
| 815 |             if immediate_burst \|\| fatigue_burst \|\| thermal_burst { | runtime gate; condition changes can en
| 816 |                 self.state.rupture_confirmed = true; | support logic; impact depends on neighboring block and d
| 817 |                 self.state.rupture_pressure_bar = Some(input.pressure_bar); | support logic; impact depends on
| 818 |                 self.state.rupture_elapsed_h = Some(self.state.total_operating_h); | support logic; impact depe
| 819 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 820 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 821 | (blank) | blank separator; no runtime behavior. |
| 822 |         let micro_area = self.spec.base_leak_area_mm2 | support logic; impact depends on neighboring block and
| 823 |             * (1.0 + 0.65 * squeeze_penalty + 0.012 * self.spec.compression_set_pct) | support logic; impact de
| 824 |             * self.spec.material.gas_permeability_factor(); | support logic; impact depends on neighboring bloc
| 825 |         let fatigue_area = self.spec.cross_section_mm * self.state.damage_index.powi(2) * 0.003; | support logi
| 826 |         let extrusion_area = self.spec.cross_section_mm * self.state.extrusion_index * 0.0025; | support logic;
| 827 |         let thermal_area = self.spec.cross_section_mm * self.state.thermal_damage * 0.0018; | support logic; im
| 828 |         let rupture_area = if self.state.rupture_confirmed { | support logic; impact depends on neighboring blo
| 829 |             if matches!(self.component, CircuitComponent::AcHose) { | runtime gate; condition changes can enabl
| 830 |                 self.spec.cross_section_mm | support logic; impact depends on neighboring block and data flow.
| 831 |                     * self.spec.cross_section_mm | support logic; impact depends on neighboring block and data
| 832 |                     * 0.14 | support logic; impact depends on neighboring block and data flow. |
| 833 |                     * self.coeffs.rupture_area_scale.max(0.05) | support logic; impact depends on neighboring b
| 834 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 835 |                 self.spec.cross_section_mm | support logic; impact depends on neighboring block and data flow.
| 836 |                     * self.spec.cross_section_mm | support logic; impact depends on neighboring block and data
| 837 |                     * 0.06 | support logic; impact depends on neighboring block and data flow. |
| 838 |                     * self.coeffs.rupture_area_scale.max(0.05) | support logic; impact depends on neighboring b
| 839 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 840 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 841 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 842 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 843 | (blank) | blank separator; no runtime behavior. |
| 844 |         self.state.leak_area_mm2 = | support logic; impact depends on neighboring block and data flow. |
| 845 |             (micro_area + fatigue_area + extrusion_area + thermal_area + rupture_area) | support logic; impact
| 846 |                 * self.seal_count.max(1) as f64; | support logic; impact depends on neighboring block and data
| 847 | (blank) | blank separator; no runtime behavior. |
| 848 |         let dp_pa = (input.delta_p_bar.max(input.pressure_bar).max(0.0)) * 1.0e5; | support logic; impact depen
| 849 |         let rho = input.fluid_density_kg_m3.max(120.0); | support logic; impact depends on neighboring block an
| 850 |         let area_m2 = self.state.leak_area_mm2 * 1.0e-6; | support logic; impact depends on neighboring block a
| 851 |         let q_m3_s = self.spec.discharge_coeff.clamp(0.05, 1.0) | support logic; impact depends on neighboring
| 852 |             * self.coeffs.flow_rate_scale.max(0.05) | support logic; impact depends on neighboring block and da
| 853 |             * area_m2 | support logic; impact depends on neighboring block and data flow. |
| 854 |             * (2.0 * dp_pa / rho).sqrt(); | support logic; impact depends on neighboring block and data flow. |
| 855 |         self.state.leak_lpm = q_m3_s * 60000.0; | support logic; impact depends on neighboring block and data f
| 856 | (blank) | blank separator; no runtime behavior. |
| 857 |         let risk = 34.0 * (pressure_ratio_max - 0.8).max(0.0) | support logic; impact depends on neighboring bl
| 858 |             + 26.0 * (pressure_ratio_rupt - 0.7).max(0.0) | support logic; impact depends on neighboring block
| 859 |             + 18.0 * self.state.damage_index | support logic; impact depends on neighboring block and data flow
| 860 |             + 12.0 * self.state.extrusion_index | support logic; impact depends on neighboring block and data f
| 861 |             + 10.0 * self.state.thermal_damage; | support logic; impact depends on neighboring block and data f
| 862 |         self.state.rupture_probability_pct = risk.clamp(0.0, 100.0); | support logic; impact depends on neighbo
| 863 | (blank) | blank separator; no runtime behavior. |
| 864 |         let rate_to_failure = damage_rate | support logic; impact depends on neighboring block and data flow. |
```

```
| 865 |             + extrusion_drive * 1.8e-4 * self.coeffs.extrusion_rate_scale.max(0.05) | support logic; impact depe
| 866 |             + thermal_rate; | support logic; impact depends on neighboring block and data flow. |
| 867 |         let pred_h = if self.state.rupture_confirmed { | support logic; impact depends on neighboring block and
| 868 |             Some(0.0) | support logic; impact depends on neighboring block and data flow. |
| 869 |         } else if rate_to_failure > 1e-8 { | support logic; impact depends on neighboring block and data flow.
| 870 |             Some(((1.0 - self.state.damage_index).max(0.0) / rate_to_failure) / 3600.0) | support logic; impact
| 871 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 872 |             None | support logic; impact depends on neighboring block and data flow. |
| 873 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 874 | (blank) | blank separator; no runtime behavior. |
| 875 |         let alert = if self.state.rupture_confirmed { | support logic; impact depends on neighboring block and
| 876 |             LeakAlertLevel::Ruptured | support logic; impact depends on neighboring block and data flow. |
| 877 |         } else if pressure_ratio_rupt > 0.9 \|\| self.state.rupture_probability_pct > 85.0 { | support logic; i
| 878 |             LeakAlertLevel::Critical | support logic; impact depends on neighboring block and data flow. |
| 879 |         } else if pressure_ratio_max > 1.0 \|\| self.state.rupture_probability_pct > 65.0 { | support logic; im
| 880 |             LeakAlertLevel::Warning | support logic; impact depends on neighboring block and data flow. |
| 881 |         } else if pressure_ratio_max > 0.85 \|\| self.state.rupture_probability_pct > 40.0 { | support logic; i
| 882 |             LeakAlertLevel::Watch | support logic; impact depends on neighboring block and data flow. |
| 883 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 884 |             LeakAlertLevel::Normal | support logic; impact depends on neighboring block and data flow. |
| 885 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 886 | (blank) | blank separator; no runtime behavior. |
| 887 |         let pressure_band = if input.pressure_bar >= self.pressure.rupture_bar { | support logic; impact depend
| 888 |             PressureBand::AtRupture | support logic; impact depends on neighboring block and data flow. |
| 889 |         } else if input.pressure_bar > self.pressure.max_bar { | support logic; impact depends on neighboring b
| 890 |             PressureBand::OverMaximum | support logic; impact depends on neighboring block and data flow. |
| 891 |         } else if input.pressure_bar < self.pressure.min_bar { | support logic; impact depends on neighboring b
| 892 |             PressureBand::BelowMinimum | support logic; impact depends on neighboring block and data flow. |
| 893 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 894 |             let span = (self.pressure.max_bar - self.pressure.min_bar).max(1e-6); | support logic; impact depen
| 895 |             let ratio = (input.pressure_bar - self.pressure.min_bar) / span; | support logic; impact depends on
| 896 |             if ratio < 0.33 { | runtime gate; condition changes can enable/disable critical paths. |
| 897 |                 PressureBand::LowBand | support logic; impact depends on neighboring block and data flow. |
| 898 |             } else if ratio < 0.66 { | support logic; impact depends on neighboring block and data flow. |
| 899 |                 PressureBand::MidBand | support logic; impact depends on neighboring block and data flow. |
| 900 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 901 |                 PressureBand::HighBand | support logic; impact depends on neighboring block and data flow. |
| 902 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 903 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 904 | (blank) | blank separator; no runtime behavior. |
| 905 |         let recommended_hold_min_bar = (self.pressure.ideal_bar * 0.65).max(self.pressure.min_bar); | support l
| 906 |         let recommended_hold_target_bar = self.pressure.ideal_bar; | support logic; impact depends on neighbori
| 907 |         let recommended_hold_max_bar = (self.pressure.max_bar * 0.88).min(self.pressure.rupture_bar * 0.78); |
| 908 |         let margin_to_max_bar = self.pressure.max_bar - input.pressure_bar; | support logic; impact depends on
| 909 |         let margin_to_rupture_bar = self.pressure.rupture_bar - input.pressure_bar; | support logic; impact depe
| 910 | (blank) | blank separator; no runtime behavior. |
| 911 |         let rca_hint = if self.state.rupture_confirmed { | support logic; impact depends on neighboring block a
| 912 |             format!( | support logic; impact depends on neighboring block and data flow. |
| 913 |                 "ruptured_at={:.2}bar; damage={:.3}; extrusion={:.3}; thermal={:.3}", | support logic; impact d
| 914 |                 self.state.rupture_pressure_bar.unwrap_or(input.pressure_bar), | support logic; impact depends
| 915 |                 self.state.damage_index, | support logic; impact depends on neighboring block and data flow. |
| 916 |                 self.state.extrusion_index, | support logic; impact depends on neighboring block and data flow.
| 917 |                 self.state.thermal_damage | support logic; impact depends on neighboring block and data flow. |
| 918 |             ) | support logic; impact depends on neighboring block and data flow. |
| 919 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 920 |             format!( | support logic; impact depends on neighboring block and data flow. |
| 921 |                 "band={}; margin_to_max={:.2}bar; margin_to_rupture={:.2}bar", | support logic; impact depends
| 922 |                 pressure_band, margin_to_max_bar, margin_to_rupture_bar | support logic; impact depends on neig
| 923 |             ) | support logic; impact depends on neighboring block and data flow. |
| 924 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 925 | (blank) | blank separator; no runtime behavior. |
| 926 |         let pca_recommendation = if self.state.rupture_confirmed { | support logic; impact depends on neighbori
| 927 |             format!( | support logic; impact depends on neighboring block and data flow. |
| 928 |                 "keep_between_{:.1}_and_{:.1}_bar; reduce_cycles_or_temp; review_material={}", | support logic;
| 929 |                 recommended_hold_min_bar, | support logic; impact depends on neighboring block and data flow. |
| 930 |                 recommended_hold_max_bar, | support logic; impact depends on neighboring block and data flow. |
| 931 |                 self.spec.material.name() | support logic; impact depends on neighboring block and data flow. |
| 932 |             ) | support logic; impact depends on neighboring block and data flow. |
| 933 |         } else if matches!(pressure_band, PressureBand::OverMaximum \| PressureBand::AtRupture) { | support log
| 934 |             format!( | support logic; impact depends on neighboring block and data flow. |
| 935 |                 "immediate_derate_to_{:.1}_bar_target_{:.1}_bar", | support logic; impact depends on neighborin
| 936 |                 recommended_hold_max_bar, recommended_hold_target_bar | support logic; impact depends on neighb
| 937 |             ) | support logic; impact depends on neighboring block and data flow. |
| 938 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 939 |             format!( | support logic; impact depends on neighboring block and data flow. |
| 940 |                 "hold_{:.1}..{:.1}_bar_target_{:.1}_bar", | support logic; impact depends on neighboring block
```

```
| 941 |                 recommended_hold_min_bar, recommended_hold_max_bar, recommended_hold_target_bar | support logic
| 942 |             ) | support logic; impact depends on neighboring block and data flow. |
| 943 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 944 | (blank) | blank separator; no runtime behavior. |
| 945 |         let warning_text = match alert { | support logic; impact depends on neighboring block and data flow. |
| 946 |             LeakAlertLevel::Ruptured => format!( | support logic; impact depends on neighboring block and data f
| 947 |                 "{}: ruptura detectada em {:.2} bar ({:.3} h); vazamento {:.2} L/min; manter {:.1}-{:.1} bar",
| 948 |                 self.application, | support logic; impact depends on neighboring block and data flow. |
| 949 |                 self.state.rupture_pressure_bar.unwrap_or(input.pressure_bar), | support logic; impact depends o
| 950 |                 self.state.rupture_elapsed_h.unwrap_or(self.state.total_operating_h), | support logic; impact de
| 951 |                 self.state.leak_lpm, | support logic; impact depends on neighboring block and data flow. |
| 952 |                 recommended_hold_min_bar, | support logic; impact depends on neighboring block and data flow. |
| 953 |                 recommended_hold_max_bar | support logic; impact depends on neighboring block and data flow. |
| 954 |             ), | support logic; impact depends on neighboring block and data flow. |
| 955 |             LeakAlertLevel::Critical => format!( | support logic; impact depends on neighboring block and data f
| 956 |                 "{}: risco critico ({:.0}%); ponto={} p={:.1}bar; margem ruptura {:.1}bar", | support logic; imp
| 957 |                 self.application, | support logic; impact depends on neighboring block and data flow. |
| 958 |                 self.state.rupture_probability_pct, | support logic; impact depends on neighboring block and dat
| 959 |                 pressure_band, | support logic; impact depends on neighboring block and data flow. |
| 960 |                 input.pressure_bar, | support logic; impact depends on neighboring block and data flow. |
| 961 |                 margin_to_rupture_bar | support logic; impact depends on neighboring block and data flow. |
| 962 |             ), | support logic; impact depends on neighboring block and data flow. |
| 963 |             LeakAlertLevel::Warning => format!( | support logic; impact depends on neighboring block and data fl
| 964 |                 "{}: sobrecarga progressiva; ponto={} p={:.1}bar; manter <= {:.1}bar", | support logic; impact d
| 965 |                 self.application, | support logic; impact depends on neighboring block and data flow. |
| 966 |                 pressure_band, | support logic; impact depends on neighboring block and data flow. |
| 967 |                 input.pressure_bar, | support logic; impact depends on neighboring block and data flow. |
| 968 |                 recommended_hold_max_bar | support logic; impact depends on neighboring block and data flow. |
| 969 |             ), | support logic; impact depends on neighboring block and data flow. |
| 970 |             LeakAlertLevel::Watch => format!( | support logic; impact depends on neighboring block and data flow
| 971 |                 "{}: degradacao progressiva; ponto={} p={:.1}bar; alvo {:.1}bar", | support logic; impact depend
| 972 |                 self.application, | support logic; impact depends on neighboring block and data flow. |
| 973 |                 pressure_band, | support logic; impact depends on neighboring block and data flow. |
| 974 |                 input.pressure_bar, | support logic; impact depends on neighboring block and data flow. |
| 975 |                 recommended_hold_target_bar | support logic; impact depends on neighboring block and data flow.
| 976 |             ), | support logic; impact depends on neighboring block and data flow. |
| 977 |             LeakAlertLevel::Normal => format!( | support logic; impact depends on neighboring block and data flo
| 978 |                 "{}: nominal; ponto={} p={:.1}bar", | support logic; impact depends on neighboring block and dat
| 979 |                 self.application, | support logic; impact depends on neighboring block and data flow. |
| 980 |                 pressure_band, | support logic; impact depends on neighboring block and data flow. |
| 981 |                 input.pressure_bar | support logic; impact depends on neighboring block and data flow. |
| 982 |             ), | support logic; impact depends on neighboring block and data flow. |
| 983 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 984 | (blank) | blank separator; no runtime behavior. |
| 985 |         CircuitResult { | support logic; impact depends on neighboring block and data flow. |
| 986 |             name: self.name.clone(), | support logic; impact depends on neighboring block and data flow. |
| 987 |             alert, | support logic; impact depends on neighboring block and data flow. |
| 988 |             pressure_band, | support logic; impact depends on neighboring block and data flow. |
| 989 |             current_pressure_bar: input.pressure_bar, | support logic; impact depends on neighboring block and d
| 990 |             rupture_probability_pct: self.state.rupture_probability_pct, | support logic; impact depends on neig
| 991 |             predicted_hours_to_rupture: pred_h, | support logic; impact depends on neighboring block and data fl
| 992 |             rupture_pressure_bar: self.state.rupture_pressure_bar, | support logic; impact depends on neighborin
| 993 |             rupture_elapsed_h: self.state.rupture_elapsed_h, | support logic; impact depends on neighboring bloc
| 994 |             leak_area_mm2: self.state.leak_area_mm2, | support logic; impact depends on neighboring block and da
| 995 |             leak_lpm: self.state.leak_lpm, | support logic; impact depends on neighboring block and data flow.
| 996 |             pressure_min_bar: self.stats.min_bar, | support logic; impact depends on neighboring block and data
| 997 |             pressure_mean_bar: self.stats.mean_bar, | support logic; impact depends on neighboring block and dat
| 998 |             pressure_max_bar: self.stats.max_bar, | support logic; impact depends on neighboring block and data
| 999 |             recommended_hold_min_bar, | support logic; impact depends on neighboring block and data flow. |
| 1000 |             recommended_hold_target_bar, | support logic; impact depends on neighboring block and data flow. |
| 1001 |             recommended_hold_max_bar, | support logic; impact depends on neighboring block and data flow. |
| 1002 |             margin_to_max_bar, | support logic; impact depends on neighboring block and data flow. |
| 1003 |             margin_to_rupture_bar, | support logic; impact depends on neighboring block and data flow. |
| 1004 |             rca_hint, | support logic; impact depends on neighboring block and data flow. |
| 1005 |             pca_recommendation, | support logic; impact depends on neighboring block and data flow. |
| 1006 |             rupture_confirmed: self.state.rupture_confirmed, | support logic; impact depends on neighboring blo
| 1007 |             warning_text, | support logic; impact depends on neighboring block and data flow. |
| 1008 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1009 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1010 | (blank) | blank separator; no runtime behavior. |
| 1011 |     pub fn reset(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1012 |         self.stats = PressureStats::new(self.pressure.mean_bar); | support logic; impact depends on neighboring
| 1013 |         self.state = SealState::new(self.spec.base_leak_area_mm2); | support logic; impact depends on neighbori
| 1014 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1015 | (blank) | blank separator; no runtime behavior. |
| 1016 |     pub fn apply_manual_params(&mut self, p: ManualCircuitParams) { | function signature/body; changes affect A
```

```
| 1017 |         if let Some(v) = p.oil_type { | runtime gate; condition changes can enable/disable critical paths. |
| 1018 |             self.oil_type = v; | support logic; impact depends on neighboring block and data flow. |
| 1019 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1020 |         if let Some(v) = p.piston_pressure_bar { | runtime gate; condition changes can enable/disable critical
| 1021 |             self.piston_pressure_bar = v.max(0.0); | support logic; impact depends on neighboring block and da
| 1022 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1023 |         if let Some(v) = p.operation_pressure_bar { | runtime gate; condition changes can enable/disable critic
| 1024 |             self.operation_pressure_bar = v.max(0.0); | support logic; impact depends on neighboring block and
| 1025 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1026 |         if let Some(v) = p.pressure_min_bar { | runtime gate; condition changes can enable/disable critical pa
| 1027 |             self.pressure.min_bar = v.max(0.0); | support logic; impact depends on neighboring block and data
| 1028 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1029 |         if let Some(v) = p.pressure_mean_bar { | runtime gate; condition changes can enable/disable critical pa
| 1030 |             self.pressure.mean_bar = v.max(0.0); | support logic; impact depends on neighboring block and data
| 1031 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1032 |         if let Some(v) = p.pressure_ideal_bar { | runtime gate; condition changes can enable/disable critical
| 1033 |             self.pressure.ideal_bar = v.max(0.0); | support logic; impact depends on neighboring block and data
| 1034 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1035 |         if let Some(v) = p.pressure_max_bar { | runtime gate; condition changes can enable/disable critical pa
| 1036 |             self.pressure.max_bar = v.max(0.0); | support logic; impact depends on neighboring block and data
| 1037 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1038 |         if let Some(v) = p.pressure_rupture_bar { | runtime gate; condition changes can enable/disable critical
| 1039 |             self.pressure.rupture_bar = v.max(self.pressure.max_bar + 0.1); | support logic; impact depends on
| 1040 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1041 |         if let Some(v) = p.oring_squeeze_pct { | runtime gate; condition changes can enable/disable critical pa
| 1042 |             self.spec.squeeze_pct = v.clamp(5.0, 45.0); | support logic; impact depends on neighboring block a
| 1043 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1044 |         if let Some(v) = p.compression_set_pct { | runtime gate; condition changes can enable/disable critical
| 1045 |             self.spec.compression_set_pct = v.clamp(0.0, 80.0); | support logic; impact depends on neighboring
| 1046 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1047 |         if let Some(v) = p.base_leak_area_mm2 { | runtime gate; condition changes can enable/disable critical
| 1048 |             self.spec.base_leak_area_mm2 = v.max(0.0); | support logic; impact depends on neighboring block an
| 1049 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1050 |         if let Some(v) = p.max_supported_temp_c { | runtime gate; condition changes can enable/disable critical
| 1051 |             let (tmin, _) = self.spec.material.temp_window_c(); | support logic; impact depends on neighboring
| 1052 |             let _ = tmin; | support logic; impact depends on neighboring block and data flow. |
| 1053 |             // user-specific temp ceiling shifts modeled via compression set drift | comment/documentation; cha
| 1054 |             let over = (self.operation_pressure_bar * 0.0 + 1.0) * (v / 180.0).clamp(0.4, 1.4); | support logic
| 1055 |             self.spec.compression_set_pct = | support logic; impact depends on neighboring block and data flow
| 1056 |                 (self.spec.compression_set_pct * (2.0 - over)).clamp(0.0, 80.0); | support logic; impact depend
| 1057 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1058 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1059 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1060 | (blank) | blank separator; no runtime behavior. |
| 1061 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 1062 | pub struct LeakPhysicsRig { | data layout contract; field changes can break callers/serde/tests. |
| 1063 |     pub circuits: Vec<LeakCircuit>, | support logic; impact depends on neighboring block and data flow. |
| 1064 |     pub alerts: Vec<LeakAlert>, | support logic; impact depends on neighboring block and data flow. |
| 1065 |     pub total_leak_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 1066 |     pub last_results: Vec<CircuitResult>, | support logic; impact depends on neighboring block and data flow.
| 1067 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1068 | (blank) | blank separator; no runtime behavior. |
| 1069 | impl Default for LeakPhysicsRig { | implementation binding; behavior semantics and trait conformance live here
| 1070 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 1071 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 1072 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1073 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1074 | (blank) | blank separator; no runtime behavior. |
| 1075 | impl LeakPhysicsRig { | implementation binding; behavior semantics and trait conformance live here. |
| 1076 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 1077 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 1078 |             circuits: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 1079 |             alerts: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 1080 |             total_leak_lpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1081 |             last_results: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 1082 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1083 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1084 | (blank) | blank separator; no runtime behavior. |
| 1085 |     pub fn with_default_machine_presets() -> Self { | function signature/body; changes affect API behavior and
| 1086 |         let mut rig = Self::new(); | support logic; impact depends on neighboring block and data flow. |
| 1087 | (blank) | blank separator; no runtime behavior. |
| 1088 |         rig.circuits.push(LeakCircuit::new( | support logic; impact depends on neighboring block and data flow
| 1089 |             "HYD_MAIN", | support logic; impact depends on neighboring block and data flow. |
| 1090 |             "O-ring circuito hidraulico principal", | support logic; impact depends on neighboring block and da
| 1091 |             CircuitComponent::Oring, | support logic; impact depends on neighboring block and data flow. |
| 1092 |             OilType::HydraulicIso46, | support logic; impact depends on neighboring block and data flow. |
```

```
| 1093 |             8, | support logic; impact depends on neighboring block and data flow. |
| 1094 |             170.0, | support logic; impact depends on neighboring block and data flow. |
| 1095 |             140.0, | support logic; impact depends on neighboring block and data flow. |
| 1096 |             PressureEnvelope { | support logic; impact depends on neighboring block and data flow. |
| 1097 |                 min_bar: 45.0, | support logic; impact depends on neighboring block and data flow. |
| 1098 |                 mean_bar: 135.0, | support logic; impact depends on neighboring block and data flow. |
| 1099 |                 ideal_bar: 160.0, | support logic; impact depends on neighboring block and data flow. |
| 1100 |                 max_bar: 230.0, | support logic; impact depends on neighboring block and data flow. |
| 1101 |                 rupture_bar: 285.0, | support logic; impact depends on neighboring block and data flow. |
| 1102 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1103 |             OringSpec { | support logic; impact depends on neighboring block and data flow. |
| 1104 |                 id_tag: "AS568-228 NBR90".to_string(), | support logic; impact depends on neighboring block an
| 1105 |                 material: OringMaterial::Nbr, | support logic; impact depends on neighboring block and data fl
| 1106 |                 shore_a: 90.0, | support logic; impact depends on neighboring block and data flow. |
| 1107 |                 cross_section_mm: 5.33, | support logic; impact depends on neighboring block and data flow. |
| 1108 |                 squeeze_pct: 18.0, | support logic; impact depends on neighboring block and data flow. |
| 1109 |                 extrusion_gap_mm: 0.20, | support logic; impact depends on neighboring block and data flow. |
| 1110 |                 compression_set_pct: 12.0, | support logic; impact depends on neighboring block and data flow.
| 1111 |                 design_life_hours: 12000.0, | support logic; impact depends on neighboring block and data flow
| 1112 |                 base_leak_area_mm2: 0.0007, | support logic; impact depends on neighboring block and data flow
| 1113 |                 discharge_coeff: 0.62, | support logic; impact depends on neighboring block and data flow. |
| 1114 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1115 |             90.0, | support logic; impact depends on neighboring block and data flow. |
| 1116 |             180.0, | support logic; impact depends on neighboring block and data flow. |
| 1117 |         )); | support logic; impact depends on neighboring block and data flow. |
| 1118 | (blank) | blank separator; no runtime behavior. |
| 1119 |         rig.circuits.push(LeakCircuit::new( | support logic; impact depends on neighboring block and data flow
| 1120 |             "ENG_OIL", | support logic; impact depends on neighboring block and data flow. |
| 1121 |             "O-ring pressao de oleo do motor", | support logic; impact depends on neighboring block and data f
| 1122 |             CircuitComponent::Seal, | support logic; impact depends on neighboring block and data flow. |
| 1123 |             OilType::Engine15w40, | support logic; impact depends on neighboring block and data flow. |
| 1124 |             3, | support logic; impact depends on neighboring block and data flow. |
| 1125 |             4.4, | support logic; impact depends on neighboring block and data flow. |
| 1126 |             3.4, | support logic; impact depends on neighboring block and data flow. |
| 1127 |             PressureEnvelope { | support logic; impact depends on neighboring block and data flow. |
| 1128 |                 min_bar: 1.2, | support logic; impact depends on neighboring block and data flow. |
| 1129 |                 mean_bar: 3.2, | support logic; impact depends on neighboring block and data flow. |
| 1130 |                 ideal_bar: 4.0, | support logic; impact depends on neighboring block and data flow. |
| 1131 |                 max_bar: 5.2, | support logic; impact depends on neighboring block and data flow. |
| 1132 |                 rupture_bar: 7.0, | support logic; impact depends on neighboring block and data flow. |
| 1133 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1134 |             OringSpec { | support logic; impact depends on neighboring block and data flow. |
| 1135 |                 id_tag: "AS568-214 FKM75".to_string(), | support logic; impact depends on neighboring block an
| 1136 |                 material: OringMaterial::Fkm, | support logic; impact depends on neighboring block and data fl
| 1137 |                 shore_a: 75.0, | support logic; impact depends on neighboring block and data flow. |
| 1138 |                 cross_section_mm: 3.53, | support logic; impact depends on neighboring block and data flow. |
| 1139 |                 squeeze_pct: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 1140 |                 extrusion_gap_mm: 0.08, | support logic; impact depends on neighboring block and data flow. |
| 1141 |                 compression_set_pct: 10.0, | support logic; impact depends on neighboring block and data flow.
| 1142 |                 design_life_hours: 10000.0, | support logic; impact depends on neighboring block and data flow
| 1143 |                 base_leak_area_mm2: 0.0003, | support logic; impact depends on neighboring block and data flow
| 1144 |                 discharge_coeff: 0.58, | support logic; impact depends on neighboring block and data flow. |
| 1145 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1146 |             28.0, | support logic; impact depends on neighboring block and data flow. |
| 1147 |             45.0, | support logic; impact depends on neighboring block and data flow. |
| 1148 |         )); | support logic; impact depends on neighboring block and data flow. |
| 1149 | (blank) | blank separator; no runtime behavior. |
| 1150 |         rig.circuits.push(LeakCircuit::new( | support logic; impact depends on neighboring block and data flow
| 1151 |             "AC_HIGH", | support logic; impact depends on neighboring block and data flow. |
| 1152 |             "O-ring alta do ar condicionado", | support logic; impact depends on neighboring block and data fl
| 1153 |             CircuitComponent::AcHose, | support logic; impact depends on neighboring block and data flow. |
| 1154 |             OilType::Pag46, | support logic; impact depends on neighboring block and data flow. |
| 1155 |             6, | support logic; impact depends on neighboring block and data flow. |
| 1156 |             16.0, | support logic; impact depends on neighboring block and data flow. |
| 1157 |             12.5, | support logic; impact depends on neighboring block and data flow. |
| 1158 |             PressureEnvelope { | support logic; impact depends on neighboring block and data flow. |
| 1159 |                 min_bar: 7.0, | support logic; impact depends on neighboring block and data flow. |
| 1160 |                 mean_bar: 12.0, | support logic; impact depends on neighboring block and data flow. |
| 1161 |                 ideal_bar: 14.0, | support logic; impact depends on neighboring block and data flow. |
| 1162 |                 max_bar: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 1163 |                 rupture_bar: 28.0, | support logic; impact depends on neighboring block and data flow. |
| 1164 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1165 |             OringSpec { | support logic; impact depends on neighboring block and data flow. |
| 1166 |                 id_tag: "AS568-210 HNBR80".to_string(), | support logic; impact depends on neighboring block a
| 1167 |                 material: OringMaterial::Hnbr, | support logic; impact depends on neighboring block and data f
| 1168 |                 shore_a: 80.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 1169 |                 cross_section_mm: 3.53, | support logic; impact depends on neighboring block and data flow. |
| 1170 |                 squeeze_pct: 17.0, | support logic; impact depends on neighboring block and data flow. |
| 1171 |                 extrusion_gap_mm: 0.10, | support logic; impact depends on neighboring block and data flow. |
| 1172 |                 compression_set_pct: 14.0, | support logic; impact depends on neighboring block and data flow.
| 1173 |                 design_life_hours: 9000.0, | support logic; impact depends on neighboring block and data flow.
| 1174 |                 base_leak_area_mm2: 0.0005, | support logic; impact depends on neighboring block and data flow
| 1175 |                 discharge_coeff: 0.55, | support logic; impact depends on neighboring block and data flow. |
| 1176 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1177 |             1.1, | support logic; impact depends on neighboring block and data flow. |
| 1178 |             12.0, | support logic; impact depends on neighboring block and data flow. |
| 1179 |         )); | support logic; impact depends on neighboring block and data flow. |
| 1180 | (blank) | blank separator; no runtime behavior. |
| 1181 |         rig | support logic; impact depends on neighboring block and data flow. |
| 1182 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1183 | (blank) | blank separator; no runtime behavior. |
| 1184 |     pub fn add_circuit(&mut self, circuit: LeakCircuit) { | function signature/body; changes affect API behavio
| 1185 |         self.circuits.push(circuit); | support logic; impact depends on neighboring block and data flow. |
| 1186 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1187 | (blank) | blank separator; no runtime behavior. |
| 1188 |     pub fn circuit_mut(&mut self, name: &str) -> Option<&mut LeakCircuit> { | function signature/body; changes
| 1189 |         self.circuits.iter_mut().find(\|c\| c.name == name) | support logic; impact depends on neighboring bloc
| 1190 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1191 | (blank) | blank separator; no runtime behavior. |
| 1192 |     pub fn oil_density_for(&self, name: &str) -> f64 { | function signature/body; changes affect API behavior a
| 1193 |         self.circuits | support logic; impact depends on neighboring block and data flow. |
| 1194 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1195 |             .find(\|c\| c.name == name) | support logic; impact depends on neighboring block and data flow. |
| 1196 |             .map(\|c\| c.oil_type.density_kg_m3()) | support logic; impact depends on neighboring block and da
| 1197 |             .unwrap_or(900.0) | support logic; impact depends on neighboring block and data flow. |
| 1198 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1199 | (blank) | blank separator; no runtime behavior. |
| 1200 |     pub fn set_runtime_pressure_state( | function signature/body; changes affect API behavior and control-flow
| 1201 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 1202 |         name: &str, | support logic; impact depends on neighboring block and data flow. |
| 1203 |         piston_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 1204 |         operation_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 1205 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 1206 |         if let Some(c) = self.circuit_mut(name) { | runtime gate; condition changes can enable/disable critical
| 1207 |             c.piston_pressure_bar = piston_pressure_bar.max(0.0); | support logic; impact depends on neighborin
| 1208 |             c.operation_pressure_bar = operation_pressure_bar.max(0.0); | support logic; impact depends on neig
| 1209 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1210 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1211 | (blank) | blank separator; no runtime behavior. |
| 1212 |     pub fn apply_manual_params(&mut self, name: &str, params: ManualCircuitParams) -> bool { | function signatu
| 1213 |         if let Some(c) = self.circuit_mut(name) { | runtime gate; condition changes can enable/disable critical
| 1214 |             c.apply_manual_params(params); | support logic; impact depends on neighboring block and data flow.
| 1215 |             true | support logic; impact depends on neighboring block and data flow. |
| 1216 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 1217 |             false | support logic; impact depends on neighboring block and data flow. |
| 1218 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1219 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1220 | (blank) | blank separator; no runtime behavior. |
| 1221 |     pub fn step(&mut self, inputs: &[(&str, CircuitInput)], dt: f64) -> Vec<CircuitResult> { | function signatu
| 1222 |         self.alerts.clear(); | support logic; impact depends on neighboring block and data flow. |
| 1223 |         self.total_leak_lpm = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1224 | (blank) | blank separator; no runtime behavior. |
| 1225 |         let mut results = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 1226 |         for (name, input) in inputs { | iteration logic; may alter timing, throughput, and safety constraints.
| 1227 |             if let Some(circuit) = self.circuits.iter_mut().find(\|c\| c.name == *name) { | runtime gate; condi
| 1228 |                 let out = circuit.step(*input, dt); | support logic; impact depends on neighboring block and da
| 1229 |                 self.total_leak_lpm += out.leak_lpm; | support logic; impact depends on neighboring block and c
| 1230 |                 if out.alert >= LeakAlertLevel::Watch { | runtime gate; condition changes can enable/disable cr
| 1231 |                     self.alerts.push(LeakAlert { | support logic; impact depends on neighboring block and data
| 1232 |                         circuit_name: out.name.clone(), | support logic; impact depends on neighboring block an
| 1233 |                         level: out.alert, | support logic; impact depends on neighboring block and data flow.
| 1234 |                         message: out.warning_text.clone(), | support logic; impact depends on neighboring bloc
| 1235 |                         predicted_hours_to_rupture: out.predicted_hours_to_rupture, | support logic; impact dep
| 1236 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 1237 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1238 |                 results.push(out); | support logic; impact depends on neighboring block and data flow. |
| 1239 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1240 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1241 | (blank) | blank separator; no runtime behavior. |
| 1242 |         self.last_results = results.clone(); | support logic; impact depends on neighboring block and data flo
| 1243 |         results | support logic; impact depends on neighboring block and data flow. |
| 1244 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 1245 | (blank) | blank separator; no runtime behavior. |
| 1246 |     pub fn reset(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1247 |         self.total_leak_lpm = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1248 |         self.alerts.clear(); | support logic; impact depends on neighboring block and data flow. |
| 1249 |         self.last_results.clear(); | support logic; impact depends on neighboring block and data flow. |
| 1250 |         for c in &mut self.circuits { | iteration logic; may alter timing, throughput, and safety constraints.
| 1251 |             c.reset(); | support logic; impact depends on neighboring block and data flow. |
| 1252 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1253 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1254 | (blank) | blank separator; no runtime behavior. |
| 1255 |     pub fn export_last_results_json<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function signatu
| 1256 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1257 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critica
| 1258 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical p
| 1259 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation sema
| 1260 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1261 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1262 |         let payload = serde_json::to_string_pretty(&self.last_results) | support logic; impact depends on neigh
| 1263 |             .map_err(\|e\| std::io::Error::other(format!("json serialization failed: {e}")))?; | error propaga
| 1264 |         fs::write(path_ref, payload) | protocol or I/O critical; changes may impact external integration corre
| 1265 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1266 | (blank) | blank separator; no runtime behavior. |
| 1267 |     pub fn export_last_results_csv<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function signatu
| 1268 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1269 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critica
| 1270 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical p
| 1271 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation sema
| 1272 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1273 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1274 |         let mut out = String::from( | support logic; impact depends on neighboring block and data flow. |
| 1275 |             "name,alert,pressure_band,current_pressure_bar,rupture_probability_pct,predicted_hours_to_rupture,
| 1276 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1277 |         for r in &self.last_results { | iteration logic; may alter timing, throughput, and safety constraints.
| 1278 |             let warn = r.warning_text.replace('"', "'").replace(',', ";"); | support logic; impact depends on
| 1279 |             let rca = r.rca_hint.replace('"', "'").replace(',', ";"); | support logic; impact depends on neig
| 1280 |             let pca = r.pca_recommendation.replace('"', "'").replace(',', ";"); | support logic; impact depen
| 1281 |             out.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 1282 |                 "{},{},{},{:.6},{:.6},{},{},{},{:.9},{:.9},{:.6},{:.6},{:.6},{:.6},{:.6},{:.6},{:.6},{:.6},\"{
| 1283 |                 r.name, | support logic; impact depends on neighboring block and data flow. |
| 1284 |                 r.alert, | support logic; impact depends on neighboring block and data flow. |
| 1285 |                 r.pressure_band, | support logic; impact depends on neighboring block and data flow. |
| 1286 |                 r.current_pressure_bar, | support logic; impact depends on neighboring block and data flow. |
| 1287 |                 r.rupture_probability_pct, | support logic; impact depends on neighboring block and data flow.
| 1288 |                 r.predicted_hours_to_rupture | support logic; impact depends on neighboring block and data flo
| 1289 |                     .map(\|v\| format!("{v:.6}")) | support logic; impact depends on neighboring block and data
| 1290 |                     .unwrap_or_default(), | support logic; impact depends on neighboring block and data flow.
| 1291 |                 r.rupture_pressure_bar | support logic; impact depends on neighboring block and data flow. |
| 1292 |                     .map(\|v\| format!("{v:.6}")) | support logic; impact depends on neighboring block and data
| 1293 |                     .unwrap_or_default(), | support logic; impact depends on neighboring block and data flow.
| 1294 |                 r.rupture_elapsed_h | support logic; impact depends on neighboring block and data flow. |
| 1295 |                     .map(\|v\| format!("{v:.6}")) | support logic; impact depends on neighboring block and data
| 1296 |                     .unwrap_or_default(), | support logic; impact depends on neighboring block and data flow.
| 1297 |                 r.leak_area_mm2, | support logic; impact depends on neighboring block and data flow. |
| 1298 |                 r.leak_lpm, | support logic; impact depends on neighboring block and data flow. |
| 1299 |                 r.pressure_min_bar, | support logic; impact depends on neighboring block and data flow. |
| 1300 |                 r.pressure_mean_bar, | support logic; impact depends on neighboring block and data flow. |
| 1301 |                 r.pressure_max_bar, | support logic; impact depends on neighboring block and data flow. |
| 1302 |                 r.recommended_hold_min_bar, | support logic; impact depends on neighboring block and data flow
| 1303 |                 r.recommended_hold_target_bar, | support logic; impact depends on neighboring block and data f
| 1304 |                 r.recommended_hold_max_bar, | support logic; impact depends on neighboring block and data flow
| 1305 |                 r.margin_to_max_bar, | support logic; impact depends on neighboring block and data flow. |
| 1306 |                 r.margin_to_rupture_bar, | support logic; impact depends on neighboring block and data flow. |
| 1307 |                 rca, | support logic; impact depends on neighboring block and data flow. |
| 1308 |                 pca, | support logic; impact depends on neighboring block and data flow. |
| 1309 |                 r.rupture_confirmed, | support logic; impact depends on neighboring block and data flow. |
| 1310 |                 warn | support logic; impact depends on neighboring block and data flow. |
| 1311 |             )); | support logic; impact depends on neighboring block and data flow. |
| 1312 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1313 |         fs::write(path_ref, out) | protocol or I/O critical; changes may impact external integration correctne
| 1314 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1315 | (blank) | blank separator; no runtime behavior. |
| 1316 |     pub fn export_predictions_json<P: AsRef<Path>>( | function signature/body; changes affect API behavior and
| 1317 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1318 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1319 |         predictions: &[ScenarioPrediction], | support logic; impact depends on neighboring block and data flow
| 1320 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
```

```
| 1321 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1322 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critical
| 1323 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical p
| 1324 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation seman
| 1325 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1326 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1327 |         let payload = serde_json::to_string_pretty(predictions) | support logic; impact depends on neighboring
| 1328 |             .map_err(\|e\| std::io::Error::other(format!("json serialization failed: {e}")))?; | error propagat
| 1329 |         fs::write(path_ref, payload) | protocol or I/O critical; changes may impact external integration correc
| 1330 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1331 | (blank) | blank separator; no runtime behavior. |
| 1332 |     pub fn export_predictions_csv<P: AsRef<Path>>( | function signature/body; changes affect API behavior and c
| 1333 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1334 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1335 |         predictions: &[ScenarioPrediction], | support logic; impact depends on neighboring block and data flow
| 1336 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1337 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1338 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critical
| 1339 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical p
| 1340 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation seman
| 1341 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1342 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1343 |         let mut out = String::from("circuit_name,scenario_name,peak_pressure_bar,final_alert,final_rupture_prob
| 1344 |         for p in predictions { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1345 |             let mode = p.likely_failure_mode.replace('"', "'").replace(',', ";"); | support logic; impact depe
| 1346 |             out.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 1347 |                 "{},{},{:.6},{},{:.6},{},\"{}\"\n", | support logic; impact depends on neighboring block and da
| 1348 |                 p.circuit_name, | support logic; impact depends on neighboring block and data flow. |
| 1349 |                 p.scenario_name, | support logic; impact depends on neighboring block and data flow. |
| 1350 |                 p.peak_pressure_bar, | support logic; impact depends on neighboring block and data flow. |
| 1351 |                 p.final_alert, | support logic; impact depends on neighboring block and data flow. |
| 1352 |                 p.final_rupture_probability_pct, | support logic; impact depends on neighboring block and data
| 1353 |                 p.hours_to_rupture | support logic; impact depends on neighboring block and data flow. |
| 1354 |                     .map(\|v\| format!("{v:.6}")) | support logic; impact depends on neighboring block and data
| 1355 |                     .unwrap_or_default(), | support logic; impact depends on neighboring block and data flow.
| 1356 |                 mode | support logic; impact depends on neighboring block and data flow. |
| 1357 |             )); | support logic; impact depends on neighboring block and data flow. |
| 1358 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1359 |         fs::write(path_ref, out) | protocol or I/O critical; changes may impact external integration correctnes
| 1360 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1361 | (blank) | blank separator; no runtime behavior. |
| 1362 |     pub fn export_material_catalog_json<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function sig
| 1363 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1364 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critical
| 1365 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical p
| 1366 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation seman
| 1367 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1368 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1369 |         let payload = serde_json::to_string_pretty(&engineering_material_catalog()) | support logic; impact dep
| 1370 |             .map_err(\|e\| std::io::Error::other(format!("json serialization failed: {e}")))?; | error propagat
| 1371 |         fs::write(path_ref, payload) | protocol or I/O critical; changes may impact external integration correc
| 1372 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1373 | (blank) | blank separator; no runtime behavior. |
| 1374 |     pub fn export_material_catalog_csv<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function sign
| 1375 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1376 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critical
| 1377 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical p
| 1378 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation seman
| 1379 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1380 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1381 |         let mut out = String::from("id,family,designation,hardness_shore,tensile_strength_mpa,elongation_pct,co
| 1382 |         for m in engineering_material_catalog() { | iteration logic; may alter timing, throughput, and safety c
| 1383 |             let notes = m.notes.replace('"', "'").replace(',', ";"); | support logic; impact depends on neighb
| 1384 |             out.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 1385 |                 "{},{},{},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},\"{}\"
| 1386 |                 m.id, | support logic; impact depends on neighboring block and data flow. |
| 1387 |                 m.family, | support logic; impact depends on neighboring block and data flow. |
| 1388 |                 m.designation, | support logic; impact depends on neighboring block and data flow. |
| 1389 |                 m.hardness_shore, | support logic; impact depends on neighboring block and data flow. |
| 1390 |                 m.tensile_strength_mpa, | support logic; impact depends on neighboring block and data flow. |
| 1391 |                 m.elongation_pct, | support logic; impact depends on neighboring block and data flow. |
| 1392 |                 m.compression_set_pct_70h, | support logic; impact depends on neighboring block and data flow.
| 1393 |                 m.tear_strength_kn_m, | support logic; impact depends on neighboring block and data flow. |
| 1394 |                 m.density_g_cm3, | support logic; impact depends on neighboring block and data flow. |
| 1395 |                 m.temp_min_c, | support logic; impact depends on neighboring block and data flow. |
| 1396 |                 m.temp_max_c, | support logic; impact depends on neighboring block and data flow. |
```

```
| 1397 |                 m.abrasion_index, | support logic; impact depends on neighboring block and data flow. |
| 1398 |                 m.corrosion_resistance_index, | support logic; impact depends on neighboring block and data flo
| 1399 |                 m.hydraulic_oil_compat, | support logic; impact depends on neighboring block and data flow. |
| 1400 |                 m.engine_oil_compat, | support logic; impact depends on neighboring block and data flow. |
| 1401 |                 m.refrigerant_oil_compat, | support logic; impact depends on neighboring block and data flow.
| 1402 |                 notes, | support logic; impact depends on neighboring block and data flow. |
| 1403 |             )); | support logic; impact depends on neighboring block and data flow. |
| 1404 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1405 |         fs::write(path_ref, out) | protocol or I/O critical; changes may impact external integration correctnes
| 1406 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1407 | (blank) | blank separator; no runtime behavior. |
| 1408 |     pub fn export_oil_catalog_json<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function signatu
| 1409 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1410 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critica
| 1411 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical |
| 1412 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation sema
| 1413 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1414 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1415 |         let payload = serde_json::to_string_pretty(&engineering_oil_catalog()) | support logic; impact depends
| 1416 |             .map_err(\|e\| std::io::Error::other(format!("json serialization failed: {e}")))?; | error propaga
| 1417 |         fs::write(path_ref, payload) | protocol or I/O critical; changes may impact external integration corre
| 1418 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1419 | (blank) | blank separator; no runtime behavior. |
| 1420 |     pub fn export_oil_catalog_csv<P: AsRef<Path>>(&self, path: P) -> std::io::Result<()> { | function signature
| 1421 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1422 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critica
| 1423 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical |
| 1424 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation sema
| 1425 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1426 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1427 |         let mut out = String::from("id,family,grade,base_stock,density_15c_kg_m3,viscosity_40c_cst,viscosity_10
| 1428 |         for o in engineering_oil_catalog() { | iteration logic; may alter timing, throughput, and safety const
| 1429 |             let notes = o.notes.replace('"', "''").replace(',', ";"); | support logic; impact depends on neighb
| 1430 |             out.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 1431 |                 "{},{},{},{},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3},{:.3
| 1432 |                 o.id, | support logic; impact depends on neighboring block and data flow. |
| 1433 |                 o.family, | support logic; impact depends on neighboring block and data flow. |
| 1434 |                 o.grade, | support logic; impact depends on neighboring block and data flow. |
| 1435 |                 o.base_stock, | support logic; impact depends on neighboring block and data flow. |
| 1436 |                 o.density_15c_kg_m3, | support logic; impact depends on neighboring block and data flow. |
| 1437 |                 o.viscosity_40c_cst, | support logic; impact depends on neighboring block and data flow. |
| 1438 |                 o.viscosity_100c_cst, | support logic; impact depends on neighboring block and data flow. |
| 1439 |                 o.viscosity_index, | support logic; impact depends on neighboring block and data flow. |
| 1440 |                 o.pour_point_c, | support logic; impact depends on neighboring block and data flow. |
| 1441 |                 o.flash_point_c, | protocol or I/O critical; changes may impact external integration correctnes
| 1442 |                 o.tan_mg_koh_g, | support logic; impact depends on neighboring block and data flow. |
| 1443 |                 o.water_content_ppm_limit, | support logic; impact depends on neighboring block and data flow.
| 1444 |                 o.demulsibility_min, | support logic; impact depends on neighboring block and data flow. |
| 1445 |                 o.copper_corrosion_level, | support logic; impact depends on neighboring block and data flow.
| 1446 |                 o.oxidation_stability_h, | support logic; impact depends on neighboring block and data flow. |
| 1447 |                 o.recommended_min_temp_c, | support logic; impact depends on neighboring block and data flow.
| 1448 |                 o.recommended_max_temp_c, | support logic; impact depends on neighboring block and data flow.
| 1449 |                 o.anti_wear_index, | support logic; impact depends on neighboring block and data flow. |
| 1450 |                 o.corrosion_inhibition_index, | support logic; impact depends on neighboring block and data flo
| 1451 |                 notes, | support logic; impact depends on neighboring block and data flow. |
| 1452 |             )); | support logic; impact depends on neighboring block and data flow. |
| 1453 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1454 |         fs::write(path_ref, out) | protocol or I/O critical; changes may impact external integration correctnes
| 1455 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1456 | (blank) | blank separator; no runtime behavior. |
| 1457 |     pub fn calibrate_from_csv<P: AsRef<Path>>( | function signature/body; changes affect API behavior and contr
| 1458 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 1459 |         csv_path: P, | support logic; impact depends on neighboring block and data flow. |
| 1460 |     ) -> std::io::Result<CalibrationReport> { | support logic; impact depends on neighboring block and data flo
| 1461 |         #[derive(Debug, Deserialize)] | comment/documentation; changing affects understanding and intent tracea
| 1462 |         struct CsvRow { | data layout contract; field changes can break callers/serde/tests. |
| 1463 |             timestamp_s: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 1464 |             circuit_name: String, | support logic; impact depends on neighboring block and data flow. |
| 1465 |             pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 1466 |             delta_p_bar: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 1467 |             temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 1468 |             cycles_per_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 1469 |             duty_01: f64, | support logic; impact depends on neighboring block and data flow. |
| 1470 |             fluid_density_kg_m3: Option<f64>, | support logic; impact depends on neighboring block and data flo
| 1471 |             measured_leak_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 1472 |             observed_rupture: Option<bool>, | support logic; impact depends on neighboring block and data flow
```

```
| 1473 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1474 | (blank) | blank separator; no runtime behavior. |
| 1475 |         let mut rdr = csv::ReaderBuilder::new() | support logic; impact depends on neighboring block and data f
| 1476 |             .has_headers(true) | support logic; impact depends on neighboring block and data flow. |
| 1477 |             .flexible(true) | support logic; impact depends on neighboring block and data flow. |
| 1478 |             .trim(csv::Trim::All) | support logic; impact depends on neighboring block and data flow. |
| 1479 |             .from_path(csv_path.as_ref()) | support logic; impact depends on neighboring block and data flow.
| 1480 |             .map_err(\|e\| std::io::Error::other(format!("open csv failed: {e}")))?; | error propagation point
| 1481 | (blank) | blank separator; no runtime behavior. |
| 1482 |         let mut samples_by_circuit: std::collections::BTreeMap<String, Vec<CalibrationCsvSample>> = | support l
| 1483 |             std::collections::BTreeMap::new(); | support logic; impact depends on neighboring block and data fl
| 1484 | (blank) | blank separator; no runtime behavior. |
| 1485 |         for rec in rdr.deserialize::<CsvRow>() { | iteration logic; may alter timing, throughput, and safety co
| 1486 |             let row = rec.map_err(\|e\| std::io::Error::other(format!("csv parse failed: {e}")))?; | error prop
| 1487 |             let circuit = row.circuit_name.trim().to_string(); | support logic; impact depends on neighboring b
| 1488 |             let density = row.fluid_density_kg_m3.unwrap_or(900.0); | support logic; impact depends on neighbor
| 1489 |             samples_by_circuit | support logic; impact depends on neighboring block and data flow. |
| 1490 |                 .entry(circuit.clone()) | support logic; impact depends on neighboring block and data flow. |
| 1491 |                 .or_default() | support logic; impact depends on neighboring block and data flow. |
| 1492 |                 .push(CalibrationCsvSample { | support logic; impact depends on neighboring block and data flow
| 1493 |                     timestamp_s: row.timestamp_s, | support logic; impact depends on neighboring block and data
| 1494 |                     circuit_name: circuit, | support logic; impact depends on neighboring block and data flow.
| 1495 |                     pressure_bar: row.pressure_bar, | support logic; impact depends on neighboring block and da
| 1496 |                     delta_p_bar: row.delta_p_bar.unwrap_or(row.pressure_bar), | support logic; impact depends o
| 1497 |                     temp_c: row.temp_c, | support logic; impact depends on neighboring block and data flow. |
| 1498 |                     cycles_per_s: row.cycles_per_s, | support logic; impact depends on neighboring block and da
| 1499 |                     duty_01: row.duty_01, | support logic; impact depends on neighboring block and data flow.
| 1500 |                     fluid_density_kg_m3: density, | support logic; impact depends on neighboring block and data
| 1501 |                     measured_leak_lpm: row.measured_leak_lpm, | support logic; impact depends on neighboring b
| 1502 |                     observed_rupture: row.observed_rupture, | support logic; impact depends on neighboring bloc
| 1503 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 1504 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1505 | (blank) | blank separator; no runtime behavior. |
| 1506 |         let mut circuit_reports = Vec::new(); | support logic; impact depends on neighboring block and data flo
| 1507 | (blank) | blank separator; no runtime behavior. |
| 1508 |         for circuit in &mut self.circuits { | iteration logic; may alter timing, throughput, and safety constra
| 1509 |             let Some(samples) = samples_by_circuit.get(&circuit.name) else { | support logic; impact depends on
| 1510 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 1511 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1512 |             if samples.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 1513 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 1514 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1515 | (blank) | blank separator; no runtime behavior. |
| 1516 |             let mut best = circuit.coeffs; | support logic; impact depends on neighboring block and data flow.
| 1517 |             let mut best_score = f64::INFINITY; | support logic; impact depends on neighboring block and data f
| 1518 | (blank) | blank separator; no runtime behavior. |
| 1519 |             let grid_damage = [0.6, 0.8, 1.0, 1.2, 1.4]; | support logic; impact depends on neighboring block a
| 1520 |             let grid_extr = [0.6, 0.85, 1.0, 1.2, 1.4]; | support logic; impact depends on neighboring block an
| 1521 |             let grid_thermal = [0.6, 0.85, 1.0, 1.2, 1.4]; | support logic; impact depends on neighboring block
| 1522 |             let grid_flow = [0.7, 0.85, 1.0, 1.15, 1.3]; | support logic; impact depends on neighboring block a
| 1523 | (blank) | blank separator; no runtime behavior. |
| 1524 |             for ds in grid_damage { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1525 |                 for es in grid_extr { | iteration logic; may alter timing, throughput, and safety constraints.
| 1526 |                     for ts in grid_thermal { | iteration logic; may alter timing, throughput, and safety const
| 1527 |                         for fs in grid_flow { | iteration logic; may alter timing, throughput, and safety cons
| 1528 |                             let mut sim = circuit.clone(); | support logic; impact depends on neighboring block
| 1529 |                             sim.reset(); | support logic; impact depends on neighboring block and data flow. |
| 1530 |                             sim.coeffs = LeakModelCoefficients { | support logic; impact depends on neighboring
| 1531 |                                 damage_rate_scale: ds, | support logic; impact depends on neighboring block an
| 1532 |                                 extrusion_rate_scale: es, | support logic; impact depends on neighboring block
| 1533 |                                 thermal_rate_scale: ts, | support logic; impact depends on neighboring block a
| 1534 |                                 flow_rate_scale: fs, | support logic; impact depends on neighboring block and
| 1535 |                                 rupture_area_scale: 1.0, | support logic; impact depends on neighboring block a
| 1536 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 1537 | (blank) | blank separator; no runtime behavior. |
| 1538 |                             let mut score = 0.0; | support logic; impact depends on neighboring block and data
| 1539 |                             let mut prev_t: Option<f64> = None; | support logic; impact depends on neighboring
| 1540 |                             for s in samples { | iteration logic; may alter timing, throughput, and safety cons
| 1541 |                                 let dt = match (prev_t, s.timestamp_s) { | support logic; impact depends on ne
| 1542 |                                     (Some(p), Some(t)) => (t - p).max(0.01), | support logic; impact depends o
| 1543 |                                     _ => 0.05, | support logic; impact depends on neighboring block and data f
| 1544 |                                 }; | scope delimiter; structural only, but wrong placement changes ownership/l
| 1545 |                                 prev_t = s.timestamp_s; | support logic; impact depends on neighboring block a
| 1546 | (blank) | blank separator; no runtime behavior. |
| 1547 |                                 let out = sim.step( | support logic; impact depends on neighboring block and da
| 1548 |                                     CircuitInput { | support logic; impact depends on neighboring block and da
```

```
| 1549 |                                        pressure_bar: s.pressure_bar, | support logic; impact depends on neighb
| 1550 |                                        delta_p_bar: s.delta_p_bar, | support logic; impact depends on neighbo
| 1551 |                                        temp_c: s.temp_c, | support logic; impact depends on neighboring block
| 1552 |                                        cycles_per_s: s.cycles_per_s, | support logic; impact depends on neighb
| 1553 |                                        duty_01: s.duty_01, | support logic; impact depends on neighboring bloc
| 1554 |                                        fluid_density_kg_m3: s.fluid_density_kg_m3, | support logic; impact dep
| 1555 |                                    }, | support logic; impact depends on neighboring block and data flow. |
| 1556 |                                    dt, | support logic; impact depends on neighboring block and data flow. |
| 1557 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 1558 | (blank) | blank separator; no runtime behavior. |
| 1559 |                                let abs_err = (out.leak_lpm - s.measured_leak_lpm).abs(); | support logic; impa
| 1560 |                                let rel = abs_err / s.measured_leak_lpm.abs().max(0.05); | support logic; impac
| 1561 |                                score += rel; | support logic; impact depends on neighboring block and data flo
| 1562 |                                if let Some(obs_rupt) = s.observed_rupture { | runtime gate; condition changes
| 1563 |                                    if obs_rupt != out.rupture_confirmed { | runtime gate; condition changes ca
| 1564 |                                        score += 2.5; | support logic; impact depends on neighboring block and
| 1565 |                                    } | scope delimiter; structural only, but wrong placement changes ownership
| 1566 |                                } | scope delimiter; structural only, but wrong placement changes ownership/li
| 1567 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 1568 | (blank) | blank separator; no runtime behavior. |
| 1569 |                            if score < best_score { | runtime gate; condition changes can enable/disable critic
| 1570 |                                best_score = score; | support logic; impact depends on neighboring block and da
| 1571 |                                best = sim.coeffs; | support logic; impact depends on neighboring block and dat
| 1572 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 1573 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1574 |                    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1575 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1576 |            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1577 | (blank) | blank separator; no runtime behavior. |
| 1578 |            circuit.coeffs = best; | support logic; impact depends on neighboring block and data flow. |
| 1579 | (blank) | blank separator; no runtime behavior. |
| 1580 |            let mut eval = circuit.clone(); | support logic; impact depends on neighboring block and data flow
| 1581 |            eval.reset(); | support logic; impact depends on neighboring block and data flow. |
| 1582 |            let mut prev_t: Option<f64> = None; | support logic; impact depends on neighboring block and data
| 1583 |            let mut se = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1584 |            let mut ape = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1585 |            let mut max_abs: f64 = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1586 |            let mut rupture_hits = 0usize; | support logic; impact depends on neighboring block and data flow.
| 1587 |            let mut rupture_total = 0usize; | support logic; impact depends on neighboring block and data flow
| 1588 |            for s in samples { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1589 |                let dt = match (prev_t, s.timestamp_s) { | support logic; impact depends on neighboring block a
| 1590 |                    (Some(p), Some(t)) => (t - p).max(0.01), | support logic; impact depends on neighboring blo
| 1591 |                    _ => 0.05, | support logic; impact depends on neighboring block and data flow. |
| 1592 |                }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1593 |                prev_t = s.timestamp_s; | support logic; impact depends on neighboring block and data flow. |
| 1594 | (blank) | blank separator; no runtime behavior. |
| 1595 |                let out = eval.step( | support logic; impact depends on neighboring block and data flow. |
| 1596 |                    CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 1597 |                        pressure_bar: s.pressure_bar, | support logic; impact depends on neighboring block and
| 1598 |                        delta_p_bar: s.delta_p_bar, | support logic; impact depends on neighboring block and da
| 1599 |                        temp_c: s.temp_c, | support logic; impact depends on neighboring block and data flow.
| 1600 |                        cycles_per_s: s.cycles_per_s, | support logic; impact depends on neighboring block and
| 1601 |                        duty_01: s.duty_01, | support logic; impact depends on neighboring block and data flow
| 1602 |                        fluid_density_kg_m3: s.fluid_density_kg_m3, | support logic; impact depends on neighbo
| 1603 |                    }, | support logic; impact depends on neighboring block and data flow. |
| 1604 |                    dt, | support logic; impact depends on neighboring block and data flow. |
| 1605 |                ); | support logic; impact depends on neighboring block and data flow. |
| 1606 |                let err = out.leak_lpm - s.measured_leak_lpm; | support logic; impact depends on neighboring b
| 1607 |                let abs_err = err.abs(); | support logic; impact depends on neighboring block and data flow. |
| 1608 |                se += err * err; | support logic; impact depends on neighboring block and data flow. |
| 1609 |                ape += abs_err / s.measured_leak_lpm.abs().max(0.05); | support logic; impact depends on neighb
| 1610 |                max_abs = max_abs.max(abs_err); | support logic; impact depends on neighboring block and data
| 1611 |                if let Some(obs) = s.observed_rupture { | runtime gate; condition changes can enable/disable c
| 1612 |                    rupture_total += 1; | support logic; impact depends on neighboring block and data flow. |
| 1613 |                    if obs == out.rupture_confirmed { | runtime gate; condition changes can enable/disable cri
| 1614 |                        rupture_hits += 1; | support logic; impact depends on neighboring block and data flow.
| 1615 |                    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1616 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1617 |            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1618 | (blank) | blank separator; no runtime behavior. |
| 1619 |            let n = samples.len() as f64; | support logic; impact depends on neighboring block and data flow.
| 1620 |            let rmse = (se / n.max(1.0)).sqrt(); | support logic; impact depends on neighboring block and data
| 1621 |            let mape = (ape / n.max(1.0)) * 100.0; | support logic; impact depends on neighboring block and dat
| 1622 |            let rupture_acc = if rupture_total > 0 { | support logic; impact depends on neighboring block and
| 1623 |                rupture_hits as f64 / rupture_total as f64 * 100.0 | support logic; impact depends on neighbor
| 1624 |            } else { | support logic; impact depends on neighboring block and data flow. |
```

```
| 1625 |                 100.0 | support logic; impact depends on neighboring block and data flow. |
| 1626 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1627 | (blank) | blank separator; no runtime behavior. |
| 1628 |             circuit_reports.push(CalibrationCircuitReport { | support logic; impact depends on neighboring bloc
| 1629 |                 circuit_name: circuit.name.clone(), | support logic; impact depends on neighboring block and da
| 1630 |                 material: circuit.spec.material.name().to_string(), | support logic; impact depends on neighbo
| 1631 |                 oil: circuit.oil_type.name().to_string(), | support logic; impact depends on neighboring block
| 1632 |                 sample_count: samples.len(), | support logic; impact depends on neighboring block and data flo
| 1633 |                 rmse_leak_lpm: rmse, | support logic; impact depends on neighboring block and data flow. |
| 1634 |                 mape_leak_pct: mape, | support logic; impact depends on neighboring block and data flow. |
| 1635 |                 max_abs_error_lpm: max_abs, | support logic; impact depends on neighboring block and data flow
| 1636 |                 rupture_accuracy_pct: rupture_acc, | support logic; impact depends on neighboring block and da
| 1637 |                 fitted: best, | support logic; impact depends on neighboring block and data flow. |
| 1638 |             }); | support logic; impact depends on neighboring block and data flow. |
| 1639 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1640 | (blank) | blank separator; no runtime behavior. |
| 1641 |         let by_material = aggregate_group_report(&circuit_reports, \|r\| r.material.clone()); | support logic;
| 1642 |         let by_oil = aggregate_group_report(&circuit_reports, \|r\| r.oil.clone()); | support logic; impact de
| 1643 |         let total_samples = circuit_reports.iter().map(\|r\| r.sample_count).sum(); | support logic; impact de
| 1644 | (blank) | blank separator; no runtime behavior. |
| 1645 |         Ok(CalibrationReport { | support logic; impact depends on neighboring block and data flow. |
| 1646 |             total_samples, | support logic; impact depends on neighboring block and data flow. |
| 1647 |             calibrated_circuits: circuit_reports.len(), | support logic; impact depends on neighboring block a
| 1648 |             circuit_reports, | support logic; impact depends on neighboring block and data flow. |
| 1649 |             by_material, | support logic; impact depends on neighboring block and data flow. |
| 1650 |             by_oil, | support logic; impact depends on neighboring block and data flow. |
| 1651 |         }) | support logic; impact depends on neighboring block and data flow. |
| 1652 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1653 | (blank) | blank separator; no runtime behavior. |
| 1654 |     pub fn export_calibration_report_json<P: AsRef<Path>>( | function signature/body; changes affect API behav
| 1655 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1656 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1657 |         report: &CalibrationReport, | support logic; impact depends on neighboring block and data flow. |
| 1658 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1659 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1660 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critica
| 1661 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical
| 1662 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation sema
| 1663 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1664 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1665 |         let payload = serde_json::to_string_pretty(report) | support logic; impact depends on neighboring bloc
| 1666 |             .map_err(\|e\| std::io::Error::other(format!("json serialization failed: {e}")))?; | error propaga
| 1667 |         fs::write(path_ref, payload) | protocol or I/O critical; changes may impact external integration corre
| 1668 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1669 | (blank) | blank separator; no runtime behavior. |
| 1670 |     pub fn export_calibration_report_csv<P: AsRef<Path>>( | function signature/body; changes affect API behavi
| 1671 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1672 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1673 |         report: &CalibrationReport, | support logic; impact depends on neighboring block and data flow. |
| 1674 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1675 |         let path_ref = path.as_ref(); | support logic; impact depends on neighboring block and data flow. |
| 1676 |         if let Some(parent) = path_ref.parent() { | runtime gate; condition changes can enable/disable critica
| 1677 |             if !parent.as_os_str().is_empty() { | runtime gate; condition changes can enable/disable critical
| 1678 |                 fs::create_dir_all(parent)?; | error propagation point; changes alter failure propagation sema
| 1679 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1680 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1681 | (blank) | blank separator; no runtime behavior. |
| 1682 |         let mut out = String::from("section,key,sample_count,rmse_leak_lpm,mape_leak_pct,max_abs_error_lpm,rup
| 1683 | (blank) | blank separator; no runtime behavior. |
| 1684 |         for r in &report.circuit_reports { | iteration logic; may alter timing, throughput, and safety constra
| 1685 |             out.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 1686 |                 "circuit,{},{},{:.6},{:.6},{:.6},{:.6},{:.6},{:.6},{:.6},{:.6},{:.6}\n", | support logic; impac
| 1687 |                 r.circuit_name, | support logic; impact depends on neighboring block and data flow. |
| 1688 |                 r.sample_count, | support logic; impact depends on neighboring block and data flow. |
| 1689 |                 r.rmse_leak_lpm, | support logic; impact depends on neighboring block and data flow. |
| 1690 |                 r.mape_leak_pct, | support logic; impact depends on neighboring block and data flow. |
| 1691 |                 r.max_abs_error_lpm, | support logic; impact depends on neighboring block and data flow. |
| 1692 |                 r.rupture_accuracy_pct, | support logic; impact depends on neighboring block and data flow. |
| 1693 |                 r.fitted.damage_rate_scale, | support logic; impact depends on neighboring block and data flow
| 1694 |                 r.fitted.extrusion_rate_scale, | support logic; impact depends on neighboring block and data f
| 1695 |                 r.fitted.thermal_rate_scale, | support logic; impact depends on neighboring block and data flo
| 1696 |                 r.fitted.flow_rate_scale, | support logic; impact depends on neighboring block and data flow.
| 1697 |                 r.fitted.rupture_area_scale, | support logic; impact depends on neighboring block and data flo
| 1698 |             )); | support logic; impact depends on neighboring block and data flow. |
| 1699 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1700 | (blank) | blank separator; no runtime behavior. |
```

```
| 1701 |         for g in &report.by_material { | iteration logic; may alter timing, throughput, and safety constraints
| 1702 |             out.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 1703 |                 "material,{},{},{:.6},{:.6},,{:.6},,,,,\n", | support logic; impact depends on neighboring bloc
| 1704 |                 g.group, | support logic; impact depends on neighboring block and data flow. |
| 1705 |                 g.sample_count, | support logic; impact depends on neighboring block and data flow. |
| 1706 |                 g.mean_rmse_lpm, | support logic; impact depends on neighboring block and data flow. |
| 1707 |                 g.mean_mape_pct, | support logic; impact depends on neighboring block and data flow. |
| 1708 |                 g.mean_rupture_accuracy_pct, | support logic; impact depends on neighboring block and data flow
| 1709 |             )); | support logic; impact depends on neighboring block and data flow. |
| 1710 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1711 | (blank) | blank separator; no runtime behavior. |
| 1712 |         for g in &report.by_oil { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1713 |             out.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 1714 |                 "oil,{},{},{:.6},{:.6},,{:.6},,,,,\n", | support logic; impact depends on neighboring block and
| 1715 |                 g.group, | support logic; impact depends on neighboring block and data flow. |
| 1716 |                 g.sample_count, | support logic; impact depends on neighboring block and data flow. |
| 1717 |                 g.mean_rmse_lpm, | support logic; impact depends on neighboring block and data flow. |
| 1718 |                 g.mean_mape_pct, | support logic; impact depends on neighboring block and data flow. |
| 1719 |                 g.mean_rupture_accuracy_pct, | support logic; impact depends on neighboring block and data flow
| 1720 |             )); | support logic; impact depends on neighboring block and data flow. |
| 1721 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1722 |         fs::write(path_ref, out) | protocol or I/O critical; changes may impact external integration correctnes
| 1723 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1724 | (blank) | blank separator; no runtime behavior. |
| 1725 |     pub fn run_monte_carlo( | function signature/body; changes affect API behavior and control-flow. |
| 1726 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1727 |         runs: usize, | support logic; impact depends on neighboring block and data flow. |
| 1728 |         horizon_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 1729 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 1730 |         variation_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 1731 |     ) -> Vec<ScenarioPrediction> { | support logic; impact depends on neighboring block and data flow. |
| 1732 |         let mut out = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 1733 |         let n = runs.max(1); | support logic; impact depends on neighboring block and data flow. |
| 1734 |         let spread = (variation_pct / 100.0).clamp(0.0, 0.95); | support logic; impact depends on neighboring
| 1735 |         for base in &self.circuits { | iteration logic; may alter timing, throughput, and safety constraints.
| 1736 |             for i in 0..n { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1737 |                 let mut c = base.clone(); | support logic; impact depends on neighboring block and data flow.
| 1738 |                 c.reset(); | support logic; impact depends on neighboring block and data flow. |
| 1739 |                 let phi = ((i as f64 * 1.618_033_988_75).fract() - 0.5) * 2.0; | support logic; impact depends
| 1740 |                 let factor = 1.0 + phi * spread; | support logic; impact depends on neighboring block and data
| 1741 |                 c.operation_pressure_bar *= factor; | support logic; impact depends on neighboring block and da
| 1742 |                 c.piston_pressure_bar *= 1.0 + phi * spread * 0.7; | support logic; impact depends on neighbori
| 1743 |                 c.spec.squeeze_pct = | support logic; impact depends on neighboring block and data flow. |
| 1744 |                     (c.spec.squeeze_pct * (1.0 - phi * spread * 0.2)).clamp(5.0, 45.0); | support logic; impact
| 1745 |                 c.spec.compression_set_pct = (c.spec.compression_set_pct | support logic; impact depends on ne
| 1746 |                     * (1.0 + phi.abs() * spread * 0.3)) | support logic; impact depends on neighboring block a
| 1747 |                     .clamp(0.0, 80.0); | support logic; impact depends on neighboring block and data flow. |
| 1748 | (blank) | blank separator; no runtime behavior. |
| 1749 |                 let mut t = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1750 |                 let mut last = CircuitResult { | support logic; impact depends on neighboring block and data f
| 1751 |                     name: c.name.clone(), | support logic; impact depends on neighboring block and data flow.
| 1752 |                     alert: LeakAlertLevel::Normal, | support logic; impact depends on neighboring block and da
| 1753 |                     pressure_band: PressureBand::MidBand, | support logic; impact depends on neighboring block
| 1754 |                     current_pressure_bar: c.pressure.mean_bar, | support logic; impact depends on neighboring
| 1755 |                     rupture_probability_pct: 0.0, | support logic; impact depends on neighboring block and data
| 1756 |                     predicted_hours_to_rupture: None, | support logic; impact depends on neighboring block and
| 1757 |                     rupture_pressure_bar: None, | support logic; impact depends on neighboring block and data
| 1758 |                     rupture_elapsed_h: None, | support logic; impact depends on neighboring block and data flow
| 1759 |                     leak_area_mm2: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1760 |                     leak_lpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1761 |                     pressure_min_bar: c.pressure.min_bar, | support logic; impact depends on neighboring block
| 1762 |                     pressure_mean_bar: c.pressure.mean_bar, | support logic; impact depends on neighboring bloc
| 1763 |                     pressure_max_bar: c.pressure.max_bar, | support logic; impact depends on neighboring block
| 1764 |                     recommended_hold_min_bar: c.pressure.ideal_bar * 0.65, | support logic; impact depends on
| 1765 |                     recommended_hold_target_bar: c.pressure.ideal_bar, | support logic; impact depends on neig
| 1766 |                     recommended_hold_max_bar: c.pressure.max_bar * 0.88, | support logic; impact depends on ne
| 1767 |                     margin_to_max_bar: c.pressure.max_bar - c.pressure.mean_bar, | support logic; impact depen
| 1768 |                     margin_to_rupture_bar: c.pressure.rupture_bar - c.pressure.mean_bar, | support logic; impac
| 1769 |                     rca_hint: String::new(), | support logic; impact depends on neighboring block and data flow
| 1770 |                     pca_recommendation: String::new(), | support logic; impact depends on neighboring block and
| 1771 |                     rupture_confirmed: false, | support logic; impact depends on neighboring block and data flo
| 1772 |                     warning_text: String::new(), | support logic; impact depends on neighboring block and data
| 1773 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1774 | (blank) | blank separator; no runtime behavior. |
| 1775 |                 while t < horizon_s { | iteration logic; may alter timing, throughput, and safety constraints.
| 1776 |                     let p = c.pressure.mean_bar * (1.0 + (t * 0.27).sin() * 0.1) * factor; | support logic; imp
```

```
| 1777 |                     last = c.step( | support logic; impact depends on neighboring block and data flow. |
| 1778 |                         CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 1779 |                             pressure_bar: p, | support logic; impact depends on neighboring block and data flo
| 1780 |                             delta_p_bar: (p - c.pressure.min_bar).max(0.0), | support logic; impact depends on
| 1781 |                             temp_c: 40.0 + (t * 0.05).sin() * 8.0, | support logic; impact depends on neighbor
| 1782 |                             cycles_per_s: 1.2 + (t * 0.11).sin().abs() * 1.8, | support logic; impact depends
| 1783 |                             duty_01: 0.55 + (t * 0.07).sin().abs() * 0.35, | support logic; impact depends on
| 1784 |                             fluid_density_kg_m3: c.oil_type.density_kg_m3(), | support logic; impact depends o
| 1785 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 1786 |                         dt.max(0.005), | support logic; impact depends on neighboring block and data flow. |
| 1787 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 1788 |                     t += dt.max(0.005); | support logic; impact depends on neighboring block and data flow. |
| 1789 |                     if last.alert == LeakAlertLevel::Ruptured { | runtime gate; condition changes can enable/d
| 1790 |                         break; | support logic; impact depends on neighboring block and data flow. |
| 1791 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1792 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1793 | (blank) | blank separator; no runtime behavior. |
| 1794 |                 out.push(ScenarioPrediction { | support logic; impact depends on neighboring block and data fl
| 1795 |                     circuit_name: c.name.clone(), | support logic; impact depends on neighboring block and data
| 1796 |                     scenario_name: format!("MC_RUN_{:04}", i), | support logic; impact depends on neighboring
| 1797 |                     peak_pressure_bar: c.stats.max_bar, | support logic; impact depends on neighboring block a
| 1798 |                     final_alert: last.alert, | support logic; impact depends on neighboring block and data flo
| 1799 |                     final_rupture_probability_pct: last.rupture_probability_pct, | support logic; impact depen
| 1800 |                     hours_to_rupture: last.predicted_hours_to_rupture, | support logic; impact depends on neig
| 1801 |                     likely_failure_mode: if last.rupture_confirmed { | support logic; impact depends on neighb
| 1802 |                         "MonteCarlo rupture".into() | support logic; impact depends on neighboring block and d
| 1803 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 1804 |                         "MonteCarlo degradation".into() | support logic; impact depends on neighboring block a
| 1805 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 1806 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 1807 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1808 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1809 |         out.sort_by(\|a, b\| { | support logic; impact depends on neighboring block and data flow. |
| 1810 |             b.final_rupture_probability_pct | support logic; impact depends on neighboring block and data flow
| 1811 |                 .total_cmp(&a.final_rupture_probability_pct) | support logic; impact depends on neighboring bl
| 1812 |         }); | support logic; impact depends on neighboring block and data flow. |
| 1813 |         out | support logic; impact depends on neighboring block and data flow. |
| 1814 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1815 | (blank) | blank separator; no runtime behavior. |
| 1816 |     pub fn predict_scenarios(&self, horizon_s: f64, dt: f64) -> Vec<ScenarioPrediction> { | function signature
| 1817 |         let scenarios: [(&str, f64, f64, f64, f64); 4] = [ | support logic; impact depends on neighboring bloc
| 1818 |             ("Nominal", 1.00, 1.00, 1.00, 1.00), | support logic; impact depends on neighboring block and data
| 1819 |             ("Sobrepressao Ciclica", 1.25, 1.15, 1.20, 1.10), | support logic; impact depends on neighboring b
| 1820 |             ("Choque Termico", 1.05, 1.35, 1.30, 1.10), | support logic; impact depends on neighboring block a
| 1821 |             ("Pico + Vibracao", 1.35, 1.10, 1.50, 1.20), | support logic; impact depends on neighboring block
| 1822 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 1823 | (blank) | blank separator; no runtime behavior. |
| 1824 |         let mut out = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 1825 |         for base in &self.circuits { | iteration logic; may alter timing, throughput, and safety constraints.
| 1826 |             for (name, p_mul, t_mul, cyc_mul, duty_mul) in scenarios { | iteration logic; may alter timing, th
| 1827 |                 let mut c = base.clone(); | support logic; impact depends on neighboring block and data flow.
| 1828 |                 c.reset(); | support logic; impact depends on neighboring block and data flow. |
| 1829 |                 let mut t = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1830 |                 let mut last = CircuitResult { | support logic; impact depends on neighboring block and data f
| 1831 |                     name: c.name.clone(), | support logic; impact depends on neighboring block and data flow.
| 1832 |                     alert: LeakAlertLevel::Normal, | support logic; impact depends on neighboring block and da
| 1833 |                     pressure_band: PressureBand::MidBand, | support logic; impact depends on neighboring block
| 1834 |                     current_pressure_bar: c.pressure.mean_bar, | support logic; impact depends on neighboring
| 1835 |                     rupture_probability_pct: 0.0, | support logic; impact depends on neighboring block and data
| 1836 |                     predicted_hours_to_rupture: None, | support logic; impact depends on neighboring block and
| 1837 |                     rupture_pressure_bar: None, | support logic; impact depends on neighboring block and data
| 1838 |                     rupture_elapsed_h: None, | support logic; impact depends on neighboring block and data flo
| 1839 |                     leak_area_mm2: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1840 |                     leak_lpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 1841 |                     pressure_min_bar: c.pressure.min_bar, | support logic; impact depends on neighboring block
| 1842 |                     pressure_mean_bar: c.pressure.mean_bar, | support logic; impact depends on neighboring bl
| 1843 |                     pressure_max_bar: c.pressure.max_bar, | support logic; impact depends on neighboring block
| 1844 |                     recommended_hold_min_bar: c.pressure.ideal_bar * 0.65, | support logic; impact depends on
| 1845 |                     recommended_hold_target_bar: c.pressure.ideal_bar, | support logic; impact depends on neig
| 1846 |                     recommended_hold_max_bar: c.pressure.max_bar * 0.88, | support logic; impact depends on ne
| 1847 |                     margin_to_max_bar: c.pressure.max_bar - c.pressure.mean_bar, | support logic; impact depen
| 1848 |                     margin_to_rupture_bar: c.pressure.rupture_bar - c.pressure.mean_bar, | support logic; impa
| 1849 |                     rca_hint: String::new(), | support logic; impact depends on neighboring block and data flo
| 1850 |                     pca_recommendation: String::new(), | support logic; impact depends on neighboring block an
| 1851 |                     rupture_confirmed: false, | support logic; impact depends on neighboring block and data fl
| 1852 |                     warning_text: String::new(), | support logic; impact depends on neighboring block and data
```

```
| 1853 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1854 | (blank) | blank separator; no runtime behavior. |
| 1855 |                 while t < horizon_s { | iteration logic; may alter timing, throughput, and safety constraints.
| 1856 |                     let phase = (t / 4.5).sin() * 0.08 + 1.0; | support logic; impact depends on neighboring bl
| 1857 |                     let p = c.pressure.mean_bar * p_mul * phase; | support logic; impact depends on neighboring
| 1858 |                     let input = CircuitInput { | support logic; impact depends on neighboring block and data fl
| 1859 |                         pressure_bar: p, | support logic; impact depends on neighboring block and data flow. |
| 1860 |                         delta_p_bar: (p - c.pressure.min_bar).max(0.0), | support logic; impact depends on neig
| 1861 |                         temp_c: 42.0 * t_mul, | support logic; impact depends on neighboring block and data flo
| 1862 |                         cycles_per_s: 1.6 * cyc_mul, | support logic; impact depends on neighboring block and
| 1863 |                         duty_01: (0.62 * duty_mul).clamp(0.0, 1.0), | support logic; impact depends on neighbo
| 1864 |                         fluid_density_kg_m3: c.oil_type.density_kg_m3(), | support logic; impact depends on ne
| 1865 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1866 |                     last = c.step(input, dt); | support logic; impact depends on neighboring block and data flo
| 1867 |                     t += dt; | support logic; impact depends on neighboring block and data flow. |
| 1868 |                     if last.alert == LeakAlertLevel::Ruptured { | runtime gate; condition changes can enable/di
| 1869 |                         break; | support logic; impact depends on neighboring block and data flow. |
| 1870 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1871 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1872 | (blank) | blank separator; no runtime behavior. |
| 1873 |                 let mode = if last.rupture_confirmed { | support logic; impact depends on neighboring block and
| 1874 |                     if matches!(c.component, CircuitComponent::AcHose) { | runtime gate; condition changes can
| 1875 |                         "Burst de mangueira A/C" | support logic; impact depends on neighboring block and data
| 1876 |                     } else if c.state.extrusion_index > c.state.thermal_damage { | support logic; impact depend
| 1877 |                         "Extrusao de vedacao" | support logic; impact depends on neighboring block and data flo
| 1878 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 1879 |                         "Ruptura por fadiga termica" | support logic; impact depends on neighboring block and
| 1880 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1881 |                 } else if last.alert >= LeakAlertLevel::Warning { | support logic; impact depends on neighbori
| 1882 |                     "Falha progressiva de vedacao" | support logic; impact depends on neighboring block and da
| 1883 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 1884 |                     "Sem ruptura no horizonte" | support logic; impact depends on neighboring block and data fl
| 1885 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1886 | (blank) | blank separator; no runtime behavior. |
| 1887 |                 out.push(ScenarioPrediction { | support logic; impact depends on neighboring block and data flo
| 1888 |                     circuit_name: c.name.clone(), | support logic; impact depends on neighboring block and data
| 1889 |                     scenario_name: name.to_string(), | support logic; impact depends on neighboring block and
| 1890 |                     peak_pressure_bar: c.stats.max_bar, | support logic; impact depends on neighboring block a
| 1891 |                     final_alert: last.alert, | support logic; impact depends on neighboring block and data flo
| 1892 |                     final_rupture_probability_pct: last.rupture_probability_pct, | support logic; impact depend
| 1893 |                     hours_to_rupture: last.predicted_hours_to_rupture, | support logic; impact depends on neigh
| 1894 |                     likely_failure_mode: mode.to_string(), | support logic; impact depends on neighboring bloc
| 1895 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 1896 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1897 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1898 | (blank) | blank separator; no runtime behavior. |
| 1899 |         out.sort_by(\|a, b\| { | support logic; impact depends on neighboring block and data flow. |
| 1900 |             b.final_rupture_probability_pct | support logic; impact depends on neighboring block and data flow
| 1901 |                 .total_cmp(&a.final_rupture_probability_pct) | support logic; impact depends on neighboring bl
| 1902 |         }); | support logic; impact depends on neighboring block and data flow. |
| 1903 |         out | support logic; impact depends on neighboring block and data flow. |
| 1904 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1905 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1906 | (blank) | blank separator; no runtime behavior. |
| 1907 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 1908 | mod tests { | module declaration; changing can break namespace graph. |
| 1909 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 1910 | (blank) | blank separator; no runtime behavior. |
| 1911 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1912 |     fn high_pressure_eventually_ruptures() { | function signature/body; changes affect API behavior and control
| 1913 |         let mut rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neighb
| 1914 |         let mut ruptured = false; | support logic; impact depends on neighboring block and data flow. |
| 1915 |         for _ in 0..4000 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 1916 |             let results = rig.step( | support logic; impact depends on neighboring block and data flow. |
| 1917 |                 &[( | support logic; impact depends on neighboring block and data flow. |
| 1918 |                     "HYD_MAIN", | support logic; impact depends on neighboring block and data flow. |
| 1919 |                     CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 1920 |                         pressure_bar: 320.0, | support logic; impact depends on neighboring block and data flow
| 1921 |                         delta_p_bar: 300.0, | support logic; impact depends on neighboring block and data flow
| 1922 |                         temp_c: 105.0, | support logic; impact depends on neighboring block and data flow. |
| 1923 |                         cycles_per_s: 3.0, | support logic; impact depends on neighboring block and data flow.
| 1924 |                         duty_01: 0.95, | support logic; impact depends on neighboring block and data flow. |
| 1925 |                         fluid_density_kg_m3: 860.0, | support logic; impact depends on neighboring block and da
| 1926 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 1927 |                 )], | support logic; impact depends on neighboring block and data flow. |
| 1928 |                 0.02, | support logic; impact depends on neighboring block and data flow. |
```

```
| 1929 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1930 |             if let Some(r) = results.first() { | runtime gate; condition changes can enable/disable critical pa
| 1931 |                 if r.rupture_confirmed { | runtime gate; condition changes can enable/disable critical paths.
| 1932 |                     ruptured = true; | support logic; impact depends on neighboring block and data flow. |
| 1933 |                     break; | support logic; impact depends on neighboring block and data flow. |
| 1934 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1935 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1936 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1937 |         assert!( | support logic; impact depends on neighboring block and data flow. |
| 1938 |             ruptured, | support logic; impact depends on neighboring block and data flow. |
| 1939 |             "expected rupture under sustained extreme overpressure" | support logic; impact depends on neighbor
| 1940 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1941 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1942 | (blank) | blank separator; no runtime behavior. |
| 1943 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1944 |     fn manual_params_are_applied() { | function signature/body; changes affect API behavior and control-flow.
| 1945 |         let mut rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neighb
| 1946 |         let ok = rig.apply_manual_params( | support logic; impact depends on neighboring block and data flow.
| 1947 |             "ENG_OIL", | support logic; impact depends on neighboring block and data flow. |
| 1948 |             ManualCircuitParams { | support logic; impact depends on neighboring block and data flow. |
| 1949 |                 oil_type: Some(OilType::Engine10w30), | support logic; impact depends on neighboring block and
| 1950 |                 piston_pressure_bar: Some(4.8), | support logic; impact depends on neighboring block and data
| 1951 |                 operation_pressure_bar: Some(3.9), | support logic; impact depends on neighboring block and dat
| 1952 |                 pressure_min_bar: Some(1.1), | support logic; impact depends on neighboring block and data flo
| 1953 |                 pressure_mean_bar: Some(3.5), | support logic; impact depends on neighboring block and data flo
| 1954 |                 pressure_ideal_bar: Some(4.4), | support logic; impact depends on neighboring block and data f
| 1955 |                 pressure_max_bar: Some(5.9), | support logic; impact depends on neighboring block and data flo
| 1956 |                 pressure_rupture_bar: Some(7.2), | support logic; impact depends on neighboring block and data
| 1957 |                 oring_squeeze_pct: Some(22.0), | support logic; impact depends on neighboring block and data f
| 1958 |                 compression_set_pct: Some(15.0), | support logic; impact depends on neighboring block and data
| 1959 |                 base_leak_area_mm2: Some(0.0004), | support logic; impact depends on neighboring block and data
| 1960 |                 max_supported_temp_c: None, | support logic; impact depends on neighboring block and data flow
| 1961 |             }, | support logic; impact depends on neighboring block and data flow. |
| 1962 |         ); | support logic; impact depends on neighboring block and data flow. |
| 1963 |         assert!(ok); | support logic; impact depends on neighboring block and data flow. |
| 1964 |         let c = rig | support logic; impact depends on neighboring block and data flow. |
| 1965 |             .circuits | support logic; impact depends on neighboring block and data flow. |
| 1966 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1967 |             .find(\|c\| c.name == "ENG_OIL") | support logic; impact depends on neighboring block and data flo
| 1968 |             .expect("circuit exists"); | panic boundary; edits alter crash behavior and resilience. |
| 1969 |         assert_eq!(c.oil_type, OilType::Engine10w30); | support logic; impact depends on neighboring block and
| 1970 |         assert!(c.pressure.max_bar > 5.8); | support logic; impact depends on neighboring block and data flow.
| 1971 |         assert!(c.spec.squeeze_pct > 21.0); | support logic; impact depends on neighboring block and data flow
| 1972 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1973 | (blank) | blank separator; no runtime behavior. |
| 1974 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1975 |     fn scenario_prediction_returns_ranked_data() { | function signature/body; changes affect API behavior and c
| 1976 |         let rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neighbori
| 1977 |         let pred = rig.predict_scenarios(180.0, 0.05); | support logic; impact depends on neighboring block and
| 1978 |         assert!(!pred.is_empty()); | support logic; impact depends on neighboring block and data flow. |
| 1979 |         assert!(pred.len() >= rig.circuits.len() * 4); | support logic; impact depends on neighboring block and
| 1980 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1981 | (blank) | blank separator; no runtime behavior. |
| 1982 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1983 |     fn monte_carlo_produces_samples() { | function signature/body; changes affect API behavior and control-flo
| 1984 |         let rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neighbori
| 1985 |         let pred = rig.run_monte_carlo(20, 120.0, 0.05, 20.0); | support logic; impact depends on neighboring
| 1986 |         assert!(pred.len() >= rig.circuits.len() * 20); | support logic; impact depends on neighboring block a
| 1987 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1988 | (blank) | blank separator; no runtime behavior. |
| 1989 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 1990 |     fn exports_json_and_csv_files() { | function signature/body; changes affect API behavior and control-flow.
| 1991 |         let mut rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neighb
| 1992 |         let _ = rig.step( | support logic; impact depends on neighboring block and data flow. |
| 1993 |             &[( | support logic; impact depends on neighboring block and data flow. |
| 1994 |                 "AC_HIGH", | support logic; impact depends on neighboring block and data flow. |
| 1995 |                 CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 1996 |                     pressure_bar: 16.0, | support logic; impact depends on neighboring block and data flow. |
| 1997 |                     delta_p_bar: 12.0, | support logic; impact depends on neighboring block and data flow. |
| 1998 |                     temp_c: 55.0, | support logic; impact depends on neighboring block and data flow. |
| 1999 |                     cycles_per_s: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 2000 |                     duty_01: 0.7, | support logic; impact depends on neighboring block and data flow. |
| 2001 |                     fluid_density_kg_m3: 995.0, | support logic; impact depends on neighboring block and data
| 2002 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 2003 |             )], | support logic; impact depends on neighboring block and data flow. |
| 2004 |             0.1, | support logic; impact depends on neighboring block and data flow. |
```

```
| 2005 |         ); | support logic; impact depends on neighboring block and data flow. |
| 2006 |         let pred = rig.predict_scenarios(60.0, 0.05); | support logic; impact depends on neighboring block and
| 2007 | (blank) | blank separator; no runtime behavior. |
| 2008 |         let root = std::env::temp_dir().join("autobreaking_leak_tests"); | support logic; impact depends on nei
| 2009 |         let runtime_json = root.join("runtime.json"); | support logic; impact depends on neighboring block and
| 2010 |         let runtime_csv = root.join("runtime.csv"); | support logic; impact depends on neighboring block and da
| 2011 |         let pred_json = root.join("pred.json"); | support logic; impact depends on neighboring block and data
| 2012 |         let pred_csv = root.join("pred.csv"); | support logic; impact depends on neighboring block and data flo
| 2013 | (blank) | blank separator; no runtime behavior. |
| 2014 |         rig.export_last_results_json(&runtime_json) | support logic; impact depends on neighboring block and da
| 2015 |             .expect("runtime json"); | panic boundary; edits alter crash behavior and resilience. |
| 2016 |         rig.export_last_results_csv(&runtime_csv) | support logic; impact depends on neighboring block and data
| 2017 |             .expect("runtime csv"); | panic boundary; edits alter crash behavior and resilience. |
| 2018 |         rig.export_predictions_json(&pred_json, &pred) | support logic; impact depends on neighboring block and
| 2019 |             .expect("pred json"); | panic boundary; edits alter crash behavior and resilience. |
| 2020 |         rig.export_predictions_csv(&pred_csv, &pred) | support logic; impact depends on neighboring block and
| 2021 |             .expect("pred csv"); | panic boundary; edits alter crash behavior and resilience. |
| 2022 | (blank) | blank separator; no runtime behavior. |
| 2023 |         assert!(runtime_json.exists()); | support logic; impact depends on neighboring block and data flow. |
| 2024 |         assert!(runtime_csv.exists()); | support logic; impact depends on neighboring block and data flow. |
| 2025 |         assert!(pred_json.exists()); | support logic; impact depends on neighboring block and data flow. |
| 2026 |         assert!(pred_csv.exists()); | support logic; impact depends on neighboring block and data flow. |
| 2027 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2028 | (blank) | blank separator; no runtime behavior. |
| 2029 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 2030 |     fn report_captures_pressure_band_and_safe_window() { | function signature/body; changes affect API behavior
| 2031 |         let mut rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neighb
| 2032 |         let out = rig.step( | support logic; impact depends on neighboring block and data flow. |
| 2033 |             &[( | support logic; impact depends on neighboring block and data flow. |
| 2034 |                 "HYD_MAIN", | support logic; impact depends on neighboring block and data flow. |
| 2035 |                 CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 2036 |                     pressure_bar: 150.0, | support logic; impact depends on neighboring block and data flow. |
| 2037 |                     delta_p_bar: 110.0, | support logic; impact depends on neighboring block and data flow. |
| 2038 |                     temp_c: 60.0, | support logic; impact depends on neighboring block and data flow. |
| 2039 |                     cycles_per_s: 1.8, | support logic; impact depends on neighboring block and data flow. |
| 2040 |                     duty_01: 0.65, | support logic; impact depends on neighboring block and data flow. |
| 2041 |                     fluid_density_kg_m3: 865.0, | support logic; impact depends on neighboring block and data
| 2042 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 2043 |             )], | support logic; impact depends on neighboring block and data flow. |
| 2044 |             0.05, | support logic; impact depends on neighboring block and data flow. |
| 2045 |         ); | support logic; impact depends on neighboring block and data flow. |
| 2046 | (blank) | blank separator; no runtime behavior. |
| 2047 |         let r = out.first().expect("result exists"); | panic boundary; edits alter crash behavior and resilienc
| 2048 |         assert!(matches!(r.pressure_band, PressureBand::LowBand \| PressureBand::MidBand \| PressureBand::High
| 2049 |         assert!(r.recommended_hold_min_bar < r.recommended_hold_target_bar); | support logic; impact depends o
| 2050 |         assert!(r.recommended_hold_target_bar < r.recommended_hold_max_bar); | support logic; impact depends o
| 2051 |         assert!(r.margin_to_rupture_bar > 0.0); | support logic; impact depends on neighboring block and data
| 2052 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2053 | (blank) | blank separator; no runtime behavior. |
| 2054 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 2055 |     fn rupture_event_records_pressure_and_time() { | function signature/body; changes affect API behavior and
| 2056 |         let mut rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neigh
| 2057 |         let mut rupture: Option<CircuitResult> = None; | support logic; impact depends on neighboring block an
| 2058 |         for _ in 0..6000 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 2059 |             let out = rig.step( | support logic; impact depends on neighboring block and data flow. |
| 2060 |                 &[( | support logic; impact depends on neighboring block and data flow. |
| 2061 |                     "HYD_MAIN", | support logic; impact depends on neighboring block and data flow. |
| 2062 |                     CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 2063 |                         pressure_bar: 330.0, | support logic; impact depends on neighboring block and data flo
| 2064 |                         delta_p_bar: 320.0, | support logic; impact depends on neighboring block and data flow
| 2065 |                         temp_c: 120.0, | support logic; impact depends on neighboring block and data flow. |
| 2066 |                         cycles_per_s: 3.8, | support logic; impact depends on neighboring block and data flow.
| 2067 |                         duty_01: 0.98, | support logic; impact depends on neighboring block and data flow. |
| 2068 |                         fluid_density_kg_m3: 860.0, | support logic; impact depends on neighboring block and da
| 2069 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 2070 |                 )], | support logic; impact depends on neighboring block and data flow. |
| 2071 |                 0.02, | support logic; impact depends on neighboring block and data flow. |
| 2072 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2073 |             if let Some(r) = out.first() { | runtime gate; condition changes can enable/disable critical paths
| 2074 |                 if r.rupture_confirmed { | runtime gate; condition changes can enable/disable critical paths.
| 2075 |                     rupture = Some(r.clone()); | support logic; impact depends on neighboring block and data f
| 2076 |                     break; | support logic; impact depends on neighboring block and data flow. |
| 2077 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2078 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2079 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2080 | (blank) | blank separator; no runtime behavior. |
```

```
| 2081 |         let r = rupture.expect("rupture expected in deterministic stress profile"); | panic boundary; edits al
| 2082 |         assert!(r.rupture_pressure_bar.is_some()); | support logic; impact depends on neighboring block and da
| 2083 |         assert!(r.rupture_elapsed_h.is_some()); | support logic; impact depends on neighboring block and data
| 2084 |         assert!(r.rupture_pressure_bar.expect("has rupture pressure") >= 280.0); | panic boundary; edits alter
| 2085 |         assert!(r.warning_text.contains("ruptura")); | support logic; impact depends on neighboring block and
| 2086 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2087 | (blank) | blank separator; no runtime behavior. |
| 2088 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 2089 |     fn engineering_catalogs_are_large_and_coherent() { | function signature/body; changes affect API behavior
| 2090 |         let mats = engineering_material_catalog(); | support logic; impact depends on neighboring block and da
| 2091 |         let oils = engineering_oil_catalog(); | support logic; impact depends on neighboring block and data fl
| 2092 |         assert!(mats.len() >= 30, "material catalog should be large"); | support logic; impact depends on neigh
| 2093 |         assert!(oils.len() >= 30, "oil catalog should be large"); | support logic; impact depends on neighborin
| 2094 | (blank) | blank separator; no runtime behavior. |
| 2095 |         let mut mat_ids: Vec<&str> = mats.iter().map(\|m\| m.id).collect(); | support logic; impact depends on
| 2096 |         mat_ids.sort_unstable(); | support logic; impact depends on neighboring block and data flow. |
| 2097 |         mat_ids.dedup(); | support logic; impact depends on neighboring block and data flow. |
| 2098 |         assert_eq!(mat_ids.len(), mats.len(), "material IDs must be unique"); | support logic; impact depends
| 2099 | (blank) | blank separator; no runtime behavior. |
| 2100 |         let mut oil_ids: Vec<&str> = oils.iter().map(\|o\| o.id).collect(); | support logic; impact depends on
| 2101 |         oil_ids.sort_unstable(); | support logic; impact depends on neighboring block and data flow. |
| 2102 |         oil_ids.dedup(); | support logic; impact depends on neighboring block and data flow. |
| 2103 |         assert_eq!(oil_ids.len(), oils.len(), "oil IDs must be unique"); | support logic; impact depends on ne
| 2104 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2105 | (blank) | blank separator; no runtime behavior. |
| 2106 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 2107 |     fn calibration_mode_ingests_csv_and_exports_reports() { | function signature/body; changes affect API beha
| 2108 |         let mut rig = LeakPhysicsRig::with_default_machine_presets(); | support logic; impact depends on neigh
| 2109 |         let root = std::env::temp_dir().join("autobreaking_calibration_tests"); | support logic; impact depend
| 2110 |         std::fs::create_dir_all(&root).expect("create calibration temp dir"); | panic boundary; edits alter cr
| 2111 |         let csv_path = root.join("bench_data.csv"); | support logic; impact depends on neighboring block and d
| 2112 | (blank) | blank separator; no runtime behavior. |
| 2113 |         let mut csv = String::from( | support logic; impact depends on neighboring block and data flow. |
| 2114 |             "timestamp_s,circuit_name,pressure_bar,delta_p_bar,temp_c,cycles_per_s,duty_01,fluid_density_kg_m3
| 2115 |         ); | support logic; impact depends on neighboring block and data flow. |
| 2116 |         for i in 0..40 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 2117 |             let t = i as f64 * 0.05; | support logic; impact depends on neighboring block and data flow. |
| 2118 |             let p = 120.0 + (i as f64 * 0.5); | support logic; impact depends on neighboring block and data fl
| 2119 |             let dp = 95.0 + (i as f64 * 0.4); | support logic; impact depends on neighboring block and data fl
| 2120 |             let temp = 65.0 + (i as f64 * 0.08); | support logic; impact depends on neighboring block and data
| 2121 |             let measured_hyd = 0.11 + (i as f64 * 0.0012); | support logic; impact depends on neighboring bloc
| 2122 |             csv.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 2123 |                 "{t:.3},HYD_MAIN,{p:.4},{dp:.4},{temp:.3},1.60,0.62,860.0,{measured_hyd:.6},false\n" | support
| 2124 |             )); | support logic; impact depends on neighboring block and data flow. |
| 2125 | (blank) | blank separator; no runtime behavior. |
| 2126 |             let p_ac = 14.0 + (i as f64 * 0.07); | support logic; impact depends on neighboring block and data
| 2127 |             let dp_ac = 10.0 + (i as f64 * 0.05); | support logic; impact depends on neighboring block and data
| 2128 |             let measured_ac = 0.018 + (i as f64 * 0.00035); | support logic; impact depends on neighboring bloc
| 2129 |             csv.push_str(&format!( | support logic; impact depends on neighboring block and data flow. |
| 2130 |                 "{t:.3},AC_HIGH,{p_ac:.4},{dp_ac:.4},48.0,1.20,0.55,995.0,{measured_ac:.6},false\n" | support
| 2131 |             )); | support logic; impact depends on neighboring block and data flow. |
| 2132 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2133 |         std::fs::write(&csv_path, csv).expect("write bench csv"); | panic boundary; edits alter crash behavior
| 2134 | (blank) | blank separator; no runtime behavior. |
| 2135 |         let report = rig.calibrate_from_csv(&csv_path).expect("run calibration"); | panic boundary; edits alte
| 2136 |         assert!(report.calibrated_circuits >= 2); | support logic; impact depends on neighboring block and data
| 2137 |         assert!(report.total_samples >= 80); | support logic; impact depends on neighboring block and data flo
| 2138 |         assert!(report.by_material.iter().any(\|g\| g.sample_count > 0)); | support logic; impact depends on n
| 2139 |         assert!(report.by_oil.iter().any(\|g\| g.sample_count > 0)); | support logic; impact depends on neighb
| 2140 | (blank) | blank separator; no runtime behavior. |
| 2141 |         let csv_report = root.join("calibration_report.csv"); | support logic; impact depends on neighboring b
| 2142 |         let json_report = root.join("calibration_report.json"); | support logic; impact depends on neighboring
| 2143 |         rig.export_calibration_report_csv(&csv_report, &report) | support logic; impact depends on neighboring
| 2144 |             .expect("export calibration csv"); | panic boundary; edits alter crash behavior and resilience. |
| 2145 |         rig.export_calibration_report_json(&json_report, &report) | support logic; impact depends on neighborin
| 2146 |             .expect("export calibration json"); | panic boundary; edits alter crash behavior and resilience. |
| 2147 | (blank) | blank separator; no runtime behavior. |
| 2148 |         assert!(csv_report.exists()); | support logic; impact depends on neighboring block and data flow. |
| 2149 |         assert!(json_report.exists()); | support logic; impact depends on neighboring block and data flow. |
| 2150 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2151 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/lib.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 1169 | Non-empty: 1081 | Symbol count: 77

## Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| new | 30 |
| tick | 18 |
| transmit | 13 |
| register_node | 9 |
| update | 8 |
| is_online | 5 |
| vehicle_heading | 5 |
| len | 4 |
| set_runtime_pressure_state | 3 |
| oil_density_for | 3 |
| default | 2 |
| is_running | 2 |
| pto_torque_demand | 2 |
| ecu_by_sa | 2 |
| run_leak_physics | 2 |
| write_f64 | 2 |
| vehicle_accel_ms2 | 2 |
| build_radar_traffic | 2 |
| build_lidar_world | 2 |
| apply_fault | 2 |
| implement_traffic | 2 |
| new_vehicle | 2 |
| kill_node | 2 |
| revive_node | 2 |
| reset | 2 |

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | module | boot_sequence | namespace and compile-time linkage |
| 9 | module | can_gateway | namespace and compile-time linkage |
| 10 | module | ecu_abs | namespace and compile-time linkage |
| 11 | module | ecu_bcm | namespace and compile-time linkage |
| 12 | module | ecu_ecm | namespace and compile-time linkage |
| 13 | module | ecu_hcm | namespace and compile-time linkage |
| 14 | module | ecu_icm | namespace and compile-time linkage |
| 15 | module | ecu_tcm | namespace and compile-time linkage |
| 16 | module | electrical | namespace and compile-time linkage |
| 17 | module | implement | namespace and compile-time linkage |
| 18 | module | j1939 | namespace and compile-time linkage |
| 19 | module | network_mgmt | namespace and compile-time linkage |
| 20 | module | nvm | namespace and compile-time linkage |
| 21 | module | uds | namespace and compile-time linkage |
| 22 | module | io | namespace and compile-time linkage |
| 25 | module | autonomous | namespace and compile-time linkage |
| 26 | module | camera | namespace and compile-time linkage |
| 27 | module | can_network | namespace and compile-time linkage |
| 28 | module | ecu_vcm | namespace and compile-time linkage |
| 29 | module | gps | namespace and compile-time linkage |
| 30 | module | imu | namespace and compile-time linkage |
| 31 | module | leak_physics | namespace and compile-time linkage |
| 32 | module | lidar | namespace and compile-time linkage |
| 33 | module | observability | namespace and compile-time linkage |
| 34 | module | radar | namespace and compile-time linkage |
| 35 | module | sim_core | namespace and compile-time linkage |
| 36 | module | v2x_telematics | namespace and compile-time linkage |
| 40 | module | adas | namespace and compile-time linkage |
| 42 | module | can_bus | namespace and compile-time linkage |
| 44 | module | chassis | namespace and compile-time linkage |
| 45 | module | ecm_heavy | namespace and compile-time linkage |
| 47 | module | engine | namespace and compile-time linkage |
| 49 | module | sensors | namespace and compile-time linkage |
| 51 | module | transmission | namespace and compile-time linkage |
| 53 | module | vehicle | namespace and compile-time linkage |
| 103 | enum | FaultType | data/schema coupling and match/serde break risk |
| 136 | impl | std::fmt::Display for FaultType | method behavior and trait semantics |

| 137 | fn | fmt | API and behavior coupling to callers/implementers |
| 168 | const | FAULT_TYPES | namespace and compile-time linkage |
| 197 | struct | HeavyMachinery | data/schema coupling and match/serde break risk |
| 254 | impl | Default for HeavyMachinery | method behavior and trait semantics |
| 255 | fn | default | API and behavior coupling to callers/implementers |
| 260 | impl | HeavyMachinery | method behavior and trait semantics |
| 261 | fn | new | API and behavior coupling to callers/implementers |
| 315 | fn | tick | API and behavior coupling to callers/implementers |
| 768 | fn | vehicle_heading | API and behavior coupling to callers/implementers |
| 771 | fn | vehicle_accel_ms2 | API and behavior coupling to callers/implementers |
| 775 | fn | run_leak_physics | API and behavior coupling to callers/implementers |
| 877 | fn | build_radar_traffic | API and behavior coupling to callers/implementers |
| 881 | fn | implement_traffic | API and behavior coupling to callers/implementers |
| 913 | fn | build_lidar_world | API and behavior coupling to callers/implementers |
| 924 | fn | apply_fault | API and behavior coupling to callers/implementers |
| 1006 | fn | inject_fault | API and behavior coupling to callers/implementers |
| 1011 | fn | clear_faults | API and behavior coupling to callers/implementers |
| 1021 | fn | key_advance | API and behavior coupling to callers/implementers |
| 1026 | fn | key_off | API and behavior coupling to callers/implementers |
| 1033 | fn | reset | API and behavior coupling to callers/implementers |
| 1037 | fn | apply_leak_manual_params | API and behavior coupling to callers/implementers |
| 1045 | fn | add_custom_leak_circuit | API and behavior coupling to callers/implementers |
| 1049 | fn | predict_leak_scenarios | API and behavior coupling to callers/implementers |
| 1053 | fn | export_leak_report_json | API and behavior coupling to callers/implementers |
| 1060 | fn | export_leak_report_csv | API and behavior coupling to callers/implementers |
| 1067 | fn | export_leak_predictions_json | API and behavior coupling to callers/implementers |
| 1075 | fn | export_leak_predictions_csv | API and behavior coupling to callers/implementers |
| 1083 | fn | export_leak_material_catalog_json | API and behavior coupling to callers/implementers |
| 1090 | fn | export_leak_material_catalog_csv | API and behavior coupling to callers/implementers |
| 1097 | fn | export_leak_oil_catalog_json | API and behavior coupling to callers/implementers |
| 1104 | fn | export_leak_oil_catalog_csv | API and behavior coupling to callers/implementers |
| 1111 | fn | calibrate_leak_model_from_csv | API and behavior coupling to callers/implementers |
| 1118 | fn | export_leak_calibration_report_json | API and behavior coupling to callers/implementers |
| 1126 | fn | export_leak_calibration_report_csv | API and behavior coupling to callers/implementers |
| 1134 | fn | monte_carlo_leak_predictions | API and behavior coupling to callers/implementers |
| 1145 | fn | export_can_snapshot_json | API and behavior coupling to callers/implementers |
| 1152 | fn | export_can_snapshot_csv | API and behavior coupling to callers/implementers |
| 1160 | fn | ignition | API and behavior coupling to callers/implementers |
| 1163 | fn | engine_running | API and behavior coupling to callers/implementers |
| 1166 | fn | all_ecus_online | API and behavior coupling to callers/implementers |

## Line by Line Change Impact

| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Heavy Machinery ECU Simulation Bench — central orchestrator. | comment/documentation; changing affects unders
| 2 | //! Instantiates every ECU module, routes J1939/CAN frames through the gateway, | comment/documentation; changing
| 3 | //! and drives the physical simulation (engine, drivetrain, hydraulics, implements). | comment/documentation; char
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use std::time::Instant; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | // ■■ Module declarations (one file per real ECU / system) ■■■■■■■■■■■■■■■■■■■■■■ | comment/documentation; cha
| 8 | pub mod boot_sequence; | module declaration; changing can break namespace graph. |
| 9 | pub mod can_gateway; | module declaration; changing can break namespace graph. |
| 10 | pub mod ecu_abs; | module declaration; changing can break namespace graph. |
| 11 | pub mod ecu_bcm; | module declaration; changing can break namespace graph. |
| 12 | pub mod ecu_ecm; | module declaration; changing can break namespace graph. |
| 13 | pub mod ecu_hcm; | module declaration; changing can break namespace graph. |
| 14 | pub mod ecu_icm; | module declaration; changing can break namespace graph. |
| 15 | pub mod ecu_tcm; | module declaration; changing can break namespace graph. |
| 16 | pub mod electrical; | module declaration; changing can break namespace graph. |
| 17 | pub mod implement; | module declaration; changing can break namespace graph. |
| 18 | pub mod j1939; | module declaration; changing can break namespace graph. |
| 19 | pub mod network_mgmt; | module declaration; changing can break namespace graph. |
| 20 | pub mod nvm; | module declaration; changing can break namespace graph. |
| 21 | pub mod uds; | module declaration; changing can break namespace graph. |
| 22 | pub mod io; // Added io module | module declaration; changing can break namespace graph. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | // ■■ Sensor suite (new) ■■■■■■■■■■■■■■■■■■■

```
| 32 | pub mod lidar; | module declaration; changing can break namespace graph. |
| 33 | pub mod observability; | module declaration; changing can break namespace graph. |
| 34 | pub mod radar; | module declaration; changing can break namespace graph. |
| 35 | pub mod sim_core; | module declaration; changing can break namespace graph. |
| 36 | pub mod v2x_telematics; | module declaration; changing can break namespace graph. |
| 37 | (blank) | blank separator; no runtime behavior. |
| 38 | // Legacy passenger-car modules (kept for reference — not used by heavy bench) | comment/documentation; changing
| 39 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 40 | mod adas; | module declaration; changing can break namespace graph. |
| 41 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 42 | mod can_bus; | module declaration; changing can break namespace graph. |
| 43 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 44 | mod chassis; | module declaration; changing can break namespace graph. |
| 45 | pub mod ecm_heavy; | module declaration; changing can break namespace graph. |
| 46 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 47 | mod engine; | module declaration; changing can break namespace graph. |
| 48 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 49 | mod sensors; | module declaration; changing can break namespace graph. |
| 50 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 51 | mod transmission; | module declaration; changing can break namespace graph. |
| 52 | #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 53 | mod vehicle; | module declaration; changing can break namespace graph. |
| 54 | (blank) | blank separator; no runtime behavior. |
| 55 | // ■■ Re-exports consumed by main.rs ■■■
```

```
| 108 |     CriticalDefLevel, // SPN 3361 FMI 1 — DEF < 2% → power derate | support logic; impact depends on neighborin
| 109 |     HighDpfSoot,      // SPN 3251 FMI 16 — DPF > 80% | support logic; impact depends on neighboring block and da
| 110 |     HighBoostPressure, // SPN 102 FMI 0 — intake manifold pressure too high | support logic; impact depends on n
| 111 |     LowBoostPressure,  // SPN 102 FMI 1 — intake manifold pressure too low | support logic; impact depends on ne
| 112 |     LowCoolantPressure, // SPN 111 FMI 1 — coolant pressure low | support logic; impact depends on neighboring b
| 113 |     HighIntakeTemp,    // SPN 105 FMI 15 — intake manifold temp high | support logic; impact depends on neighbor
| 114 |     HighExhaustTemp,   // SPN 173 FMI 0 — exhaust temperature high | support logic; impact depends on neighborin
| 115 |     PoorDefQuality,    // SPN 1761 FMI 3 — DEF quality/sensor abnormal | support logic; impact depends on neighb
| 116 |     HighNoxTailpipe,   // SPN 3226 FMI 16 — tailpipe NOx above threshold | support logic; impact depends on neig
| 117 |     LowFuelRailPressure, // SPN 157 FMI 1 — common-rail pressure low | support logic; impact depends on neighbor
| 118 |     // Transmission faults | comment/documentation; changing affects understanding and intent traceability. |
| 119 |     TransClutchOverheat, // clutch temp > 200°C | support logic; impact depends on neighboring block and data fl
| 120 |     TransNeutralFailure, // TCM cannot engage neutral | support logic; impact depends on neighboring block and d
| 121 |     // Electrical faults | comment/documentation; changing affects understanding and intent traceability. |
| 122 |     LowBatteryVoltage, // battery < 11V | support logic; impact depends on neighboring block and data flow. |
| 123 |     AlternatorFault,   // alternator < 12V while engine running | support logic; impact depends on neighboring b
| 124 |     // Hydraulic faults | comment/documentation; changing affects understanding and intent traceability. |
| 125 |     HighHydTemp,     // hydraulic fluid > 100°C | support logic; impact depends on neighboring block and data fl
| 126 |     LowHydLevel,     // hydraulic level < 20% | support logic; impact depends on neighboring block and data flow
| 127 |     HydFilterBypass, // filter ΔP > 6 bar | support logic; impact depends on neighboring block and data flow. |
| 128 |     // CAN faults | comment/documentation; changing affects understanding and intent traceability. |
| 129 |     BusOffInjection, // force bus-off condition | support logic; impact depends on neighboring block and data fl
| 130 |     KillEcm,         // power off ECM node | support logic; impact depends on neighboring block and data flow.
| 131 |     KillTcm,         // power off TCM node | support logic; impact depends on neighboring block and data flow.
| 132 |     // None selected | comment/documentation; changing affects understanding and intent traceability. |
| 133 |     None, | support logic; impact depends on neighboring block and data flow. |
| 134 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 135 | (blank) | blank separator; no runtime behavior. |
| 136 | impl std::fmt::Display for FaultType { | implementation binding; behavior semantics and trait conformance live i
| 137 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 138 |         let s = match self { | support logic; impact depends on neighboring block and data flow. |
| 139 |             FaultType::HighCoolantTemp => "High Coolant Temp (SPN110 FMI0)", | support logic; impact depends on
| 140 |             FaultType::LowOilPressure => "Low Oil Pressure (SPN100 FMI1)", | support logic; impact depends on ne
| 141 |             FaultType::LowFuelPressure => "Low Fuel Pressure (SPN94 FMI1)", | support logic; impact depends on n
| 142 |             FaultType::CriticalDefLevel => "Critical DEF Level (SPN3361 FMI1)", | support logic; impact depends
| 143 |             FaultType::HighDpfSoot => "High DPF Soot (SPN3251 FMI16)", | support logic; impact depends on neighb
| 144 |             FaultType::HighBoostPressure => "High Boost Pressure (SPN102 FMI0)", | support logic; impact depends
| 145 |             FaultType::LowBoostPressure => "Low Boost Pressure (SPN102 FMI1)", | support logic; impact depends o
| 146 |             FaultType::LowCoolantPressure => "Low Coolant Pressure (SPN111 FMI1)", | support logic; impact depen
| 147 |             FaultType::HighIntakeTemp => "High Intake Temp (SPN105 FMI15)", | support logic; impact depends on n
| 148 |             FaultType::HighExhaustTemp => "High Exhaust Temp (SPN173 FMI0)", | support logic; impact depends on
| 149 |             FaultType::PoorDefQuality => "Poor DEF Quality (SPN1761 FMI3)", | support logic; impact depends on n
| 150 |             FaultType::HighNoxTailpipe => "High Tailpipe NOx (SPN3226 FMI16)", | support logic; impact depends o
| 151 |             FaultType::LowFuelRailPressure => "Low Fuel Rail Pressure (SPN157 FMI1)", | support logic; impact de
| 152 |             FaultType::TransClutchOverheat => "Trans Clutch Overheat", | support logic; impact depends on neighb
| 153 |             FaultType::TransNeutralFailure => "Trans Neutral Failure", | support logic; impact depends on neighb
| 154 |             FaultType::LowBatteryVoltage => "Low Battery Voltage", | support logic; impact depends on neighborin
| 155 |             FaultType::AlternatorFault => "Alternator Fault", | support logic; impact depends on neighboring blo
| 156 |             FaultType::HighHydTemp => "Hydraulic Fluid High Temp", | support logic; impact depends on neighborin
| 157 |             FaultType::LowHydLevel => "Hydraulic Level Low", | support logic; impact depends on neighboring bloc
| 158 |             FaultType::HydFilterBypass => "Hyd Filter Bypass Open", | support logic; impact depends on neighbori
| 159 |             FaultType::BusOffInjection => "CAN Bus-Off Injection", | support logic; impact depends on neighborin
| 160 |             FaultType::KillEcm => "Kill ECM Node", | support logic; impact depends on neighboring block and data
| 161 |             FaultType::KillTcm => "Kill TCM Node", | support logic; impact depends on neighboring block and data
| 162 |             FaultType::None => "(no fault selected)", | support logic; impact depends on neighboring block and d
| 163 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 164 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 165 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 166 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 167 | (blank) | blank separator; no runtime behavior. |
| 168 | pub const FAULT_TYPES: &[FaultType] = &[ | support logic; impact depends on neighboring block and data flow. |
| 169 |     FaultType::None, | support logic; impact depends on neighboring block and data flow. |
| 170 |     FaultType::HighCoolantTemp, | support logic; impact depends on neighboring block and data flow. |
| 171 |     FaultType::LowOilPressure, | support logic; impact depends on neighboring block and data flow. |
| 172 |     FaultType::LowFuelPressure, | support logic; impact depends on neighboring block and data flow. |
| 173 |     FaultType::CriticalDefLevel, | support logic; impact depends on neighboring block and data flow. |
| 174 |     FaultType::HighDpfSoot, | support logic; impact depends on neighboring block and data flow. |
| 175 |     FaultType::HighBoostPressure, | support logic; impact depends on neighboring block and data flow. |
| 176 |     FaultType::LowBoostPressure, | support logic; impact depends on neighboring block and data flow. |
| 177 |     FaultType::LowCoolantPressure, | support logic; impact depends on neighboring block and data flow. |
| 178 |     FaultType::HighIntakeTemp, | support logic; impact depends on neighboring block and data flow. |
| 179 |     FaultType::HighExhaustTemp, | support logic; impact depends on neighboring block and data flow. |
| 180 |     FaultType::PoorDefQuality, | support logic; impact depends on neighboring block and data flow. |
| 181 |     FaultType::HighNoxTailpipe, | support logic; impact depends on neighboring block and data flow. |
| 182 |     FaultType::LowFuelRailPressure, | support logic; impact depends on neighboring block and data flow. |
| 183 |     FaultType::TransClutchOverheat, | support logic; impact depends on neighboring block and data flow. |
```

```
| 184 |     FaultType::TransNeutralFailure, | support logic; impact depends on neighboring block and data flow. |
| 185 |     FaultType::LowBatteryVoltage, | support logic; impact depends on neighboring block and data flow. |
| 186 |     FaultType::AlternatorFault, | support logic; impact depends on neighboring block and data flow. |
| 187 |     FaultType::HighHydTemp, | support logic; impact depends on neighboring block and data flow. |
| 188 |     FaultType::LowHydLevel, | support logic; impact depends on neighboring block and data flow. |
| 189 |     FaultType::HydFilterBypass, | support logic; impact depends on neighboring block and data flow. |
| 190 |     FaultType::BusOffInjection, | support logic; impact depends on neighboring block and data flow. |
| 191 |     FaultType::KillEcm, | support logic; impact depends on neighboring block and data flow. |
| 192 |     FaultType::KillTcm, | support logic; impact depends on neighboring block and data flow. |
| 193 | ]; | support logic; impact depends on neighboring block and data flow. |
| 194 | (blank) | blank separator; no runtime behavior. |
| 195 | // ■■ Top-level Heavy Machinery Bench ■■■
```

```
| 260 | impl HeavyMachinery { | implementation binding; behavior semantics and trait conformance live here. |
| 261 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 262 |         let mut gw = CanGateway::new(); | support logic; impact depends on neighboring block and data flow. |
| 263 |         gw.register_node(j1939::addr::ECM_1, "ECM #1"); | support logic; impact depends on neighboring block and
| 264 |         gw.register_node(j1939::addr::TRANSMISSION, "TCM"); | support logic; impact depends on neighboring bloc
| 265 |         gw.register_node(j1939::addr::CAB, "BCM/CAB"); | support logic; impact depends on neighboring block and
| 266 |         gw.register_node(j1939::addr::INSTRUMENT, "ICM/DASH"); | support logic; impact depends on neighboring b
| 267 |         gw.register_node(j1939::addr::HITCH, "HCM/HITCH"); | support logic; impact depends on neighboring block
| 268 |         gw.register_node(j1939::addr::BRAKES, "ABS/ESP"); | support logic; impact depends on neighboring block
| 269 |         gw.register_node(j1939::addr::ISOBUS_VT, "ISOBUS-VT"); | support logic; impact depends on neighboring b
| 270 |         gw.register_node(j1939::addr::TASK_CTRL, "ISOBUS-TC"); | support logic; impact depends on neighboring b
| 271 |         gw.register_node(j1939::addr::IMPLEMENT, "IMPLEMENT"); | support logic; impact depends on neighboring b
| 272 | (blank) | blank separator; no runtime behavior. |
| 273 |         HeavyMachinery { | support logic; impact depends on neighboring block and data flow. |
| 274 |             gateway: gw, | support logic; impact depends on neighboring block and data flow. |
| 275 |             boot: BootSequence::new(), | support logic; impact depends on neighboring block and data flow. |
| 276 |             net_mgmt: NetworkManager::new(), | support logic; impact depends on neighboring block and data flow
| 277 |             ecm: EcuEcm::new(), | support logic; impact depends on neighboring block and data flow. |
| 278 |             tcm: EcuTcm::new(), | support logic; impact depends on neighboring block and data flow. |
| 279 |             bcm: EcuBcm::new(), | support logic; impact depends on neighboring block and data flow. |
| 280 |             icm: EcuIcm::new(), | support logic; impact depends on neighboring block and data flow. |
| 281 |             hcm: EcuHcm::new(), | support logic; impact depends on neighboring block and data flow. |
| 282 |             abs: EcuAbs::new(), | support logic; impact depends on neighboring block and data flow. |
| 283 |             electrical: ElectricalSystem::new(), | support logic; impact depends on neighboring block and data
| 284 |             uds_ecm: UdsServer::new("ECM #1", j1939::addr::ECM_1), | protocol or I/O critical; changes may impac
| 285 |             uds_tcm: UdsServer::new("TCM", j1939::addr::TRANSMISSION), | protocol or I/O critical; changes may
| 286 |             nvm: NvmStore::new(), | support logic; impact depends on neighboring block and data flow. |
| 287 |             implement: ImplementControl::new(), | support logic; impact depends on neighboring block and data f
| 288 |             elapsed: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 289 |             throttle_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 290 |             brake_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 291 |             hitch_joystick: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 292 |             loader_lift_cmd: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 293 |             loader_tilt_cmd: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 294 |             selected_fault: FaultType::None, | support logic; impact depends on neighboring block and data flow
| 295 |             fault_active: false, | support logic; impact depends on neighboring block and data flow. |
| 296 |             gps: GpsModule::new(), | support logic; impact depends on neighboring block and data flow. |
| 297 |             imu: Imu::new(), | support logic; impact depends on neighboring block and data flow. |
| 298 |             radar: RadarSuite::new(), | support logic; impact depends on neighboring block and data flow. |
| 299 |             lidar: LidarSensor::new_vlp16(), | support logic; impact depends on neighboring block and data flow
| 300 |             ad: AutonomousController::new(), | support logic; impact depends on neighboring block and data flow
| 301 |             v2x: V2xModule::new(), | support logic; impact depends on neighboring block and data flow. |
| 302 |             telematics: TelematicsModule::new(), | support logic; impact depends on neighboring block and data
| 303 |             can_net: CanNetwork::new(), | support logic; impact depends on neighboring block and data flow. |
| 304 |             vcm: EcuVcm::new(), | support logic; impact depends on neighboring block and data flow. |
| 305 |             camera: CameraSystem::new(), | support logic; impact depends on neighboring block and data flow. |
| 306 |             fusion: SensorFusion::new(), | support logic; impact depends on neighboring block and data flow. |
| 307 |             leak_rig: LeakPhysicsRig::with_default_machine_presets(), | support logic; impact depends on neighb
| 308 |             leak_reports: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 309 |             metrics: SimMetrics::default(), | support logic; impact depends on neighboring block and data flow.
| 310 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 311 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 312 | (blank) | blank separator; no runtime behavior. |
| 313 |     // ═══════════════════════════════════════════════════════════════════════════════ | comment/d
| 314 |     /// Master simulation tick — called every 16 ms (≈ 60 Hz). | comment/documentation; changing affects underst
| 315 |     pub fn tick(&mut self, dt: f64) { | function signature/body; changes affect API behavior and control-flow.
| 316 |         let step_started = Instant::now(); | support logic; impact depends on neighboring block and data flow.
| 317 |         let body_load_requested_a = { | support logic; impact depends on neighboring block and data flow. |
| 318 |             let mut load = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 319 |             if self.bcm.work_lights_front { | runtime gate; condition changes can enable/disable critical paths
| 320 |                 load += 18.0; | support logic; impact depends on neighboring block and data flow. |
| 321 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 322 |             if self.bcm.work_lights_rear { | runtime gate; condition changes can enable/disable critical paths.
| 323 |                 load += 14.0; | support logic; impact depends on neighboring block and data flow. |
| 324 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 325 |             if self.bcm.road_lights { | runtime gate; condition changes can enable/disable critical paths. |
| 326 |                 load += 8.0; | support logic; impact depends on neighboring block and data flow. |
| 327 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 328 |             if self.bcm.beacon_light { | runtime gate; condition changes can enable/disable critical paths. |
| 329 |                 load += 4.0; | support logic; impact depends on neighboring block and data flow. |
| 330 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 331 |             if self.bcm.hazard_lights { | runtime gate; condition changes can enable/disable critical paths. |
| 332 |                 load += 6.0; | support logic; impact depends on neighboring block and data flow. |
| 333 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 334 |             if self.bcm.cab_interior_light { | runtime gate; condition changes can enable/disable critical paths
| 335 |                 load += 2.0; | support logic; impact depends on neighboring block and data flow. |
```

```
| 336 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 337 |             if self.bcm.hvac_on { | runtime gate; condition changes can enable/disable critical paths. |
| 338 |                 load += 15.0; | support logic; impact depends on neighboring block and data flow. |
| 339 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 |             if self.bcm.horn_active { | runtime gate; condition changes can enable/disable critical paths. |
| 341 |                 load += 8.0; | support logic; impact depends on neighboring block and data flow. |
| 342 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 343 |             load | support logic; impact depends on neighboring block and data flow. |
| 344 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 345 |         let electrical_loads = ElectricalLoadRequest { | support logic; impact depends on neighboring block and
| 346 |             bcm_supply_a: 5.0 + body_load_requested_a, | support logic; impact depends on neighboring block and
| 347 |             vcm_supply_a: 4.0, | support logic; impact depends on neighboring block and data flow. |
| 348 |             ecm_supply_a: if matches!(self.boot.ignition, IgnitionState::On \| IgnitionState::Cranking \| Ignit
| 349 |             tcm_supply_a: if matches!(self.boot.ignition, IgnitionState::On \| IgnitionState::Cranking \| Ignit
| 350 |             abs_supply_a: if matches!(self.boot.ignition, IgnitionState::On \| IgnitionState::Cranking \| Ignit
| 351 |             hcm_supply_a: if matches!(self.boot.ignition, IgnitionState::On \| IgnitionState::Cranking \| Ignit
| 352 |             icm_supply_a: if matches!(self.boot.ignition, IgnitionState::Accessory \| IgnitionState::On \| Igni
| 353 |             body_load_a: body_load_requested_a, | support logic; impact depends on neighboring block and data f
| 354 |             hs_can_a: 0.15, | support logic; impact depends on neighboring block and data flow. |
| 355 |             ms_can_a: 0.08, | support logic; impact depends on neighboring block and data flow. |
| 356 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 357 |         self.electrical.tick( | support logic; impact depends on neighboring block and data flow. |
| 358 |             self.boot.ignition, | support logic; impact depends on neighboring block and data flow. |
| 359 |             self.bcm.battery_voltage, | support logic; impact depends on neighboring block and data flow. |
| 360 |             self.ecm.alternator_v, | support logic; impact depends on neighboring block and data flow. |
| 361 |             &electrical_loads, | support logic; impact depends on neighboring block and data flow. |
| 362 |             &self.bcm.fuses, | support logic; impact depends on neighboring block and data flow. |
| 363 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 364 |         ); | support logic; impact depends on neighboring block and data flow. |
| 365 |         let powered_sas = self.electrical.powered_sas(); | support logic; impact depends on neighboring block a
| 366 | (blank) | blank separator; no runtime behavior. |
| 367 |         // 1. ■■ Boot sequence ■■■
```

```
| 412 |                 self.throttle_pct | support logic; impact depends on neighboring block and data flow. |
| 413 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 414 |             self.ecm.tick(thr, total_engine_load, dt) | support logic; impact depends on neighboring block and d
| 415 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 416 |             self.ecm.rpm = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 417 |             self.ecm.actual_torque_nm = 0.0; | support logic; impact depends on neighboring block and data flow
| 418 |             self.ecm.alternator_v = self.bcm.battery_voltage; | support logic; impact depends on neighboring blo
| 419 |             Vec::new() | support logic; impact depends on neighboring block and data flow. |
| 420 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 421 | (blank) | blank separator; no runtime behavior. |
| 422 |         // 4. ■■ TCM ■■■
```

```
| 488 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 489 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 490 |         ); | support logic; impact depends on neighboring block and data flow. |
| 491 |         let _effective_throttle = self.throttle_pct * (1.0 - tcs_cut); | support logic; impact depends on neighb
| 492 | (blank) | blank separator; no runtime behavior. |
| 493 |         // 10. ■■ UDS diagnostic server ticks ■■■
```

```
| 564 |         self.gps | support logic; impact depends on neighboring block and data flow. |
| 565 |             .update(self.tcm.ground_speed_kmh, self.vehicle_heading(), dt); | support logic; impact depends on n
| 566 | (blank) | blank separator; no runtime behavior. |
| 567 |         // IMU | comment/documentation; changing affects understanding and intent traceability. |
| 568 |         self.imu.update( | support logic; impact depends on neighboring block and data flow. |
| 569 |             self.vehicle_accel_ms2(), | support logic; impact depends on neighboring block and data flow. |
| 570 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 571 |             self.tcm.ground_speed_kmh * 0.01, | support logic; impact depends on neighboring block and data flow
| 572 |             self.vehicle_heading(), | support logic; impact depends on neighboring block and data flow. |
| 573 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 574 |             self.ecm.rpm, | support logic; impact depends on neighboring block and data flow. |
| 575 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 576 |         ); | support logic; impact depends on neighboring block and data flow. |
| 577 | (blank) | blank separator; no runtime behavior. |
| 578 |         // RADAR — build traffic objects from implement traffic simulation | comment/documentation; changing aff
| 579 |         let traffic = self.build_radar_traffic(); | support logic; impact depends on neighboring block and data
| 580 |         self.radar.update(spd_ms, &traffic, dt); | support logic; impact depends on neighboring block and data f
| 581 | (blank) | blank separator; no runtime behavior. |
| 582 |         // LIDAR — build world obstacles and scan | comment/documentation; changing affects understanding and in
| 583 |         let world = self.build_lidar_world(); | support logic; impact depends on neighboring block and data flow
| 584 |         self.lidar.update(&world, self.imu.pitch_deg, dt); | support logic; impact depends on neighboring block
| 585 | (blank) | blank separator; no runtime behavior. |
| 586 |         // Camera + Sensor Fusion | comment/documentation; changing affects understanding and intent traceabilit
| 587 |         self.camera.update( | support logic; impact depends on neighboring block and data flow. |
| 588 |             self.tcm.ground_speed_kmh, | support logic; impact depends on neighboring block and data flow. |
| 589 |             self.vehicle_heading(), | support logic; impact depends on neighboring block and data flow. |
| 590 |             self.radar.ttc_front, | support logic; impact depends on neighboring block and data flow. |
| 591 |             self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 592 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 593 |         ); | support logic; impact depends on neighboring block and data flow. |
| 594 |         self.fusion | support logic; impact depends on neighboring block and data flow. |
| 595 |             .fuse(&self.camera, &self.radar.front_center.targets, spd_ms, dt); | support logic; impact depends o
| 596 | (blank) | blank separator; no runtime behavior. |
| 597 |         // AD controller | comment/documentation; changing affects understanding and intent traceability. |
| 598 |         let lane_off = self.imu.lateral_g * 0.2; // rough lane offset from lateral G | support logic; impact dep
| 599 |         let lane_conf = if self.tcm.ground_speed_kmh > 5.0 { | support logic; impact depends on neighboring bloc
| 600 |             0.92 | support logic; impact depends on neighboring block and data flow. |
| 601 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 602 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 603 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 604 |         let fused_ttc = self.fusion.ttc_critical.min(self.radar.ttc_front); | support logic; impact depends on n
| 605 |         let fused_lead = self.fusion.lead_dist_m.min(self.radar.closest_front_m); | support logic; impact depend
| 606 |         let ttc = fused_ttc; | support logic; impact depends on neighboring block and data flow. |
| 607 |         let lead_spd = if self.radar.front_center.closest_inpath().is_some() { | support logic; impact depends o
| 608 |             (self.tcm.ground_speed_kmh - 5.0).max(0.0) / 3.6 | support logic; impact depends on neighboring bloc
| 609 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 610 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 611 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 612 |         let (ad_thr, ad_brk, ad_str) = self.ad.tick( | support logic; impact depends on neighboring block and da
| 613 |             self.tcm.ground_speed_kmh, | support logic; impact depends on neighboring block and data flow. |
| 614 |             self.throttle_pct / 100.0, | support logic; impact depends on neighboring block and data flow. |
| 615 |             self.brake_pct / 100.0, | support logic; impact depends on neighboring block and data flow. |
| 616 |             fused_lead, | support logic; impact depends on neighboring block and data flow. |
| 617 |             lead_spd, | support logic; impact depends on neighboring block and data flow. |
| 618 |             lane_off, | support logic; impact depends on neighboring block and data flow. |
| 619 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 620 |             lane_conf, | support logic; impact depends on neighboring block and data flow. |
| 621 |             ttc, | support logic; impact depends on neighboring block and data flow. |
| 622 |             self.radar.bsm_left, | support logic; impact depends on neighboring block and data flow. |
| 623 |             self.radar.bsm_right, | support logic; impact depends on neighboring block and data flow. |
| 624 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 625 |         ); | support logic; impact depends on neighboring block and data flow. |
| 626 |         // AD overrides apply back to vehicle (if engaged) | comment/documentation; changing affects understandi
| 627 |         if self.ad.engaged { | runtime gate; condition changes can enable/disable critical paths. |
| 628 |             if let Some(v) = ad_thr { | runtime gate; condition changes can enable/disable critical paths. |
| 629 |                 self.throttle_pct = (v * 100.0).clamp(0.0, 100.0); | support logic; impact depends on neighborin
| 630 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 631 |             if let Some(v) = ad_brk { | runtime gate; condition changes can enable/disable critical paths. |
| 632 |                 self.brake_pct = (v * 100.0).clamp(0.0, 100.0); | support logic; impact depends on neighboring b
| 633 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 634 |             if let Some(v) = ad_str { | runtime gate; condition changes can enable/disable critical paths. |
| 635 |                 self.ad.steer_cmd_deg = v; | support logic; impact depends on neighboring block and data flow.
| 636 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 637 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 638 | (blank) | blank separator; no runtime behavior. |
| 639 |         // V2X | comment/documentation; changing affects understanding and intent traceability. |
```

```
| 640 |         self.v2x.update( | support logic; impact depends on neighboring block and data flow. |
| 641 |             self.gps.latitude_deg, | support logic; impact depends on neighboring block and data flow. |
| 642 |             self.gps.longitude_deg, | support logic; impact depends on neighboring block and data flow. |
| 643 |             spd_ms, | support logic; impact depends on neighboring block and data flow. |
| 644 |             self.vehicle_heading(), | support logic; impact depends on neighboring block and data flow. |
| 645 |             self.imu.longitudinal_g * 9.81, | support logic; impact depends on neighboring block and data flow.
| 646 |             self.imu.lateral_g * 9.81, | support logic; impact depends on neighboring block and data flow. |
| 647 |             self.abs.abs_system_active, | support logic; impact depends on neighboring block and data flow. |
| 648 |             self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 649 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 650 |         ); | support logic; impact depends on neighboring block and data flow. |
| 651 | (blank) | blank separator; no runtime behavior. |
| 652 |         // Telematics | comment/documentation; changing affects understanding and intent traceability. |
| 653 |         self.telematics.update( | support logic; impact depends on neighboring block and data flow. |
| 654 |             self.gps.latitude_deg, | support logic; impact depends on neighboring block and data flow. |
| 655 |             self.gps.longitude_deg, | support logic; impact depends on neighboring block and data flow. |
| 656 |             self.tcm.ground_speed_kmh, | support logic; impact depends on neighboring block and data flow. |
| 657 |             self.ecm.engine_hours, | support logic; impact depends on neighboring block and data flow. |
| 658 |             self.ecm.active_dtcs.len(), | support logic; impact depends on neighboring block and data flow. |
| 659 |             self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 660 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 661 |         ); | support logic; impact depends on neighboring block and data flow. |
| 662 | (blank) | blank separator; no runtime behavior. |
| 663 |         // VCM | comment/documentation; changing affects understanding and intent traceability. |
| 664 |         let abs_torque_limit_nm = if self.abs.abs_system_active \|\| self.abs.tcs_system_active { | support logi
| 665 |             260.0 | support logic; impact depends on neighboring block and data flow. |
| 666 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 667 |             1050.0 | support logic; impact depends on neighboring block and data flow. |
| 668 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 669 |         let ad_torque_limit_nm = if fused_ttc < 1.2 { | support logic; impact depends on neighboring block and
| 670 |             0.0 | support logic; impact depends on neighboring block and data flow. |
| 671 |         } else if fused_ttc < 2.0 { | support logic; impact depends on neighboring block and data flow. |
| 672 |             220.0 | support logic; impact depends on neighboring block and data flow. |
| 673 |         } else if fused_ttc < 3.5 { | support logic; impact depends on neighboring block and data flow. |
| 674 |             520.0 | support logic; impact depends on neighboring block and data flow. |
| 675 |         } else if fused_lead < 12.0 { | support logic; impact depends on neighboring block and data flow. |
| 676 |             680.0 | support logic; impact depends on neighboring block and data flow. |
| 677 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 678 |             1050.0 | support logic; impact depends on neighboring block and data flow. |
| 679 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 680 |         let vcm_frames = self.vcm.tick( | support logic; impact depends on neighboring block and data flow. |
| 681 |             self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 682 |             self.ecm.rpm, | support logic; impact depends on neighboring block and data flow. |
| 683 |             self.tcm.ground_speed_kmh, | support logic; impact depends on neighboring block and data flow. |
| 684 |             self.ecm.actual_torque_nm, | support logic; impact depends on neighboring block and data flow. |
| 685 |             self.hcm.hydraulic_power_kw, | support logic; impact depends on neighboring block and data flow. |
| 686 |             self.implement.pto_torque_demand(), | support logic; impact depends on neighboring block and data f
| 687 |             self.bcm.total_load_amps, | support logic; impact depends on neighboring block and data flow. |
| 688 |             abs_torque_limit_nm, | support logic; impact depends on neighboring block and data flow. |
| 689 |             ad_torque_limit_nm, | support logic; impact depends on neighboring block and data flow. |
| 690 |             self.ecm.active_dtcs.len() as u32, | support logic; impact depends on neighboring block and data flo
| 691 |             &self.gateway.dispatched.clone(), | support logic; impact depends on neighboring block and data flow
| 692 |             dt, | support logic; impact depends on neighboring block and data flow. |
| 693 |         ); | support logic; impact depends on neighboring block and data flow. |
| 694 |         for f in vcm_frames { | iteration logic; may alter timing, throughput, and safety constraints. |
| 695 |             self.gateway.transmit(f); | support logic; impact depends on neighboring block and data flow. |
| 696 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 697 | (blank) | blank separator; no runtime behavior. |
| 698 |         // ADAS/fusion/energy status frames on the powertrain network via VCM SA. | comment/documentation; chang
| 699 |         let lead_distance = self.fusion.lead_dist_m.min(self.radar.closest_front_m); | support logic; impact dep
| 700 |         let critical_ttc = self.fusion.ttc_critical.min(self.radar.ttc_front); | support logic; impact depends o
| 701 |         let in_path_hazard = self | support logic; impact depends on neighboring block and data flow. |
| 702 |             .fusion | support logic; impact depends on neighboring block and data flow. |
| 703 |             .objects | support logic; impact depends on neighboring block and data flow. |
| 704 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 705 |             .any(\|o\| o.in_path && o.ttc_s < 3.0); | support logic; impact depends on neighboring block and da
| 706 |         let speed_limit = self.camera.active_speed_limit.unwrap_or(0) as f64; | support logic; impact depends o
| 707 |         let adas = j1939::Builder::adas1( | support logic; impact depends on neighboring block and data flow. |
| 708 |             self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 709 |             self.ad.lka_state == FeatureState::Active, | support logic; impact depends on neighboring block and
| 710 |             self.ad.acc_state == FeatureState::Active, | support logic; impact depends on neighboring block and
| 711 |             self.ad.aeb_state == FeatureState::Active, | support logic; impact depends on neighboring block and
| 712 |             lead_distance, | support logic; impact depends on neighboring block and data flow. |
| 713 |             critical_ttc, | support logic; impact depends on neighboring block and data flow. |
| 714 |             speed_limit, | support logic; impact depends on neighboring block and data flow. |
| 715 |             j1939::addr::HEADWAY, | support logic; impact depends on neighboring block and data flow. |
```

```
| 716 |         ); | support logic; impact depends on neighboring block and data flow. |
| 717 |         let fus = j1939::Builder::fus1( | support logic; impact depends on neighboring block and data flow. |
| 718 |             self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 719 |             self.fusion.objects.len(), | support logic; impact depends on neighboring block and data flow. |
| 720 |             critical_ttc, | support logic; impact depends on neighboring block and data flow. |
| 721 |             self.fusion | support logic; impact depends on neighboring block and data flow. |
| 722 |                 .objects | support logic; impact depends on neighboring block and data flow. |
| 723 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 724 |                 .map(\|o\| o.confidence) | support logic; impact depends on neighboring block and data flow. |
| 725 |                 .fold(0.0, f64::max), | support logic; impact depends on neighboring block and data flow. |
| 726 |             in_path_hazard, | support logic; impact depends on neighboring block and data flow. |
| 727 |             j1939::addr::HEADWAY, | support logic; impact depends on neighboring block and data flow. |
| 728 |         ); | support logic; impact depends on neighboring block and data flow. |
| 729 |         let engy = j1939::Builder::engy1( | support logic; impact depends on neighboring block and data flow. |
| 730 |             self.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 731 |             self.bcm.battery_voltage, | support logic; impact depends on neighboring block and data flow. |
| 732 |             self.bcm.total_load_amps, | support logic; impact depends on neighboring block and data flow. |
| 733 |             self.bcm.total_load_amps * self.bcm.battery_voltage / 1000.0, | support logic; impact depends on nei
| 734 |             self.hcm.hydraulic_power_kw, | support logic; impact depends on neighboring block and data flow. |
| 735 |             j1939::addr::HEADWAY, | support logic; impact depends on neighboring block and data flow. |
| 736 |         ); | support logic; impact depends on neighboring block and data flow. |
| 737 |         self.gateway.transmit(adas); | support logic; impact depends on neighboring block and data flow. |
| 738 |         self.gateway.transmit(fus); | support logic; impact depends on neighboring block and data flow. |
| 739 |         self.gateway.transmit(engy); | support logic; impact depends on neighboring block and data flow. |
| 740 | (blank) | blank separator; no runtime behavior. |
| 741 |         // 17. ■■ Active fault injection ■■■
```

```
| 792 |             (self.ecm.oil_pressure_kpa / 100.0).max(0.0), | support logic; impact depends on neighboring block a
| 793 |             (self.ecm.oil_pressure_kpa / 100.0).max(0.0), | support logic; impact depends on neighboring block a
| 794 |         ); | support logic; impact depends on neighboring block and data flow. |
| 795 |         self.leak_rig | support logic; impact depends on neighboring block and data flow. |
| 796 |             .set_runtime_pressure_state("AC_HIGH", ac_high_bar, ac_high_bar); | support logic; impact depends o
| 797 | (blank) | blank separator; no runtime behavior. |
| 798 |         let hyd_rho = self.leak_rig.oil_density_for("HYD_MAIN"); | support logic; impact depends on neighboring
| 799 |         let eng_rho = self.leak_rig.oil_density_for("ENG_OIL"); | support logic; impact depends on neighboring b
| 800 |         let ac_rho = self.leak_rig.oil_density_for("AC_HIGH"); | support logic; impact depends on neighboring bl
| 801 | (blank) | blank separator; no runtime behavior. |
| 802 |         let inputs = [ | support logic; impact depends on neighboring block and data flow. |
| 803 |             ( | support logic; impact depends on neighboring block and data flow. |
| 804 |                 "HYD_MAIN", | support logic; impact depends on neighboring block and data flow. |
| 805 |                 CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 806 |                     pressure_bar: self.hcm.system_pressure_bar, | support logic; impact depends on neighboring b
| 807 |                     delta_p_bar: (self.hcm.system_pressure_bar - self.hcm.return_pres_bar).max(0.0), | support l
| 808 |                     temp_c: self.hcm.fluid_temp_c, | support logic; impact depends on neighboring block and data
| 809 |                     cycles_per_s: (self.hcm.total_flow_demand_lpm / 45.0).clamp(0.0, 5.0), | support logic; impa
| 810 |                     duty_01: (self.hcm.total_flow_demand_lpm / self.hcm.pump_flow_lpm.max(1.0)) | support logic
| 811 |                         .clamp(0.0, 1.0), | support logic; impact depends on neighboring block and data flow. |
| 812 |                     fluid_density_kg_m3: hyd_rho, | support logic; impact depends on neighboring block and data
| 813 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 814 |             ), | support logic; impact depends on neighboring block and data flow. |
| 815 |             ( | support logic; impact depends on neighboring block and data flow. |
| 816 |                 "ENG_OIL", | support logic; impact depends on neighboring block and data flow. |
| 817 |                 CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 818 |                     pressure_bar: (self.ecm.oil_pressure_kpa / 100.0).max(0.0), | support logic; impact depends
| 819 |                     delta_p_bar: (self.ecm.oil_pressure_kpa / 100.0).max(0.0), | support logic; impact depends o
| 820 |                     temp_c: self.ecm.oil_temp_c, | support logic; impact depends on neighboring block and data f
| 821 |                     cycles_per_s: (self.ecm.rpm / 60.0).clamp(0.0, 60.0), | support logic; impact depends on nei
| 822 |                     duty_01: (self.ecm.active_throttle / 100.0).clamp(0.0, 1.0), | support logic; impact depends
| 823 |                     fluid_density_kg_m3: eng_rho, | support logic; impact depends on neighboring block and data
| 824 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 825 |             ), | support logic; impact depends on neighboring block and data flow. |
| 826 |             ( | support logic; impact depends on neighboring block and data flow. |
| 827 |                 "AC_HIGH", | support logic; impact depends on neighboring block and data flow. |
| 828 |                 CircuitInput { | support logic; impact depends on neighboring block and data flow. |
| 829 |                     pressure_bar: ac_high_bar, | support logic; impact depends on neighboring block and data flo
| 830 |                     delta_p_bar: (ac_high_bar - 2.0).max(0.0), | support logic; impact depends on neighboring bl
| 831 |                     temp_c: self.bcm.cab_temp_c + 8.0, | support logic; impact depends on neighboring block and
| 832 |                     cycles_per_s: (self.ecm.rpm / 120.0).clamp(0.0, 30.0), | support logic; impact depends on ne
| 833 |                     duty_01: hvac_duty, | support logic; impact depends on neighboring block and data flow. |
| 834 |                     fluid_density_kg_m3: ac_rho, | support logic; impact depends on neighboring block and data f
| 835 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 836 |             ), | support logic; impact depends on neighboring block and data flow. |
| 837 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 838 | (blank) | blank separator; no runtime behavior. |
| 839 |         let reports = self.leak_rig.step(&inputs, dt); | support logic; impact depends on neighboring block and
| 840 |         self.leak_reports = reports.clone(); | support logic; impact depends on neighboring block and data flow
| 841 | (blank) | blank separator; no runtime behavior. |
| 842 |         for rep in &reports { | iteration logic; may alter timing, throughput, and safety constraints. |
| 843 |             let leaked_l = rep.leak_lpm * dt / 60.0; | support logic; impact depends on neighboring block and da
| 844 |             match rep.name.as_str() { | branch dispatcher; arm changes alter behavior class handling. |
| 845 |                 "HYD_MAIN" => { | support logic; impact depends on neighboring block and data flow. |
| 846 |                     let pressure_drop_bar = leaked_l * 85.0; | support logic; impact depends on neighboring bloc
| 847 |                     self.hcm.system_pressure_bar = | support logic; impact depends on neighboring block and data
| 848 |                         (self.hcm.system_pressure_bar - pressure_drop_bar).max(0.0); | support logic; impact dep
| 849 |                     self.hcm.fluid_level_pct = | support logic; impact depends on neighboring block and data flo
| 850 |                         (self.hcm.fluid_level_pct - leaked_l / 90.0 * 100.0).clamp(0.0, 100.0); | support logic
| 851 |                     if rep.alert >= LeakAlertLevel::Warning { | runtime gate; condition changes can enable/disa
| 852 |                         self.hcm.alarm_low_pressure = true; | support logic; impact depends on neighboring bloc
| 853 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 854 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 855 |                 "ENG_OIL" => { | support logic; impact depends on neighboring block and data flow. |
| 856 |                     let pressure_scale = (1.0 - (rep.leak_lpm / 12.0)).clamp(0.12, 1.0); | support logic; impac
| 857 |                     self.ecm.oil_pressure_kpa *= pressure_scale; | support logic; impact depends on neighboring
| 858 |                     self.ecm.oil_level_pct = | support logic; impact depends on neighboring block and data flow
| 859 |                         (self.ecm.oil_level_pct - leaked_l / 28.0 * 100.0).clamp(0.0, 100.0); | support logic; i
| 860 |                     if rep.alert >= LeakAlertLevel::Warning && self.ecm.rpm > 600.0 { | runtime gate; condition
| 861 |                         self.ecm.amber_lamp = true; | support logic; impact depends on neighboring block and da
| 862 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 863 |                     if rep.alert >= LeakAlertLevel::Critical && self.ecm.rpm > 600.0 { | runtime gate; conditio
| 864 |                         self.ecm.red_lamp = true; | support logic; impact depends on neighboring block and data
| 865 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 866 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 867 |                 "AC_HIGH" | support logic; impact depends on neighboring block and data flow. |
```

```
| 868 |                     if self.bcm.hvac_on => { | runtime gate; condition changes can enable/disable critical paths
| 869 |                         let cooling_loss = (rep.leak_lpm / 1.5).clamp(0.0, 1.0); | support logic; impact depends
| 870 |                         self.bcm.cab_temp_c += (0.22 + cooling_loss * 0.55) * dt; | support logic; impact depend
| 871 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 872 |                 _ => {} | support logic; impact depends on neighboring block and data flow. |
| 873 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 874 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 875 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 876 | (blank) | blank separator; no runtime behavior. |
| 877 |     fn build_radar_traffic(&self) -> Vec<SimTrafficObj> { | function signature/body; changes affect API behavior
| 878 |         self.implement_traffic() | support logic; impact depends on neighboring block and data flow. |
| 879 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 880 | (blank) | blank separator; no runtime behavior. |
| 881 |     fn implement_traffic(&self) -> Vec<SimTrafficObj> { | function signature/body; changes affect API behavior a
| 882 |         use radar::RadarObjectClass; | import wiring; changing path/name can break compilation and module linkag
| 883 |         let t = self.elapsed; | support logic; impact depends on neighboring block and data flow. |
| 884 |         let lead_dist = 50.0 + (t * 0.1).sin() * 30.0; | support logic; impact depends on neighboring block and
| 885 |         vec![ | support logic; impact depends on neighboring block and data flow. |
| 886 |             SimTrafficObj { | support logic; impact depends on neighboring block and data flow. |
| 887 |                 id: 0, | support logic; impact depends on neighboring block and data flow. |
| 888 |                 distance_m: lead_dist.max(5.0), | support logic; impact depends on neighboring block and data f
| 889 |                 lateral_offset_m: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 890 |                 speed_ms: 10.0 + (t * 0.05).sin() * 5.0, | support logic; impact depends on neighboring block a
| 891 |                 rcs_dbsm: 12.0, | support logic; impact depends on neighboring block and data flow. |
| 892 |                 object_type: RadarObjectClass::Car, | support logic; impact depends on neighboring block and da
| 893 |             }, | support logic; impact depends on neighboring block and data flow. |
| 894 |             SimTrafficObj { | support logic; impact depends on neighboring block and data flow. |
| 895 |                 id: 1, | support logic; impact depends on neighboring block and data flow. |
| 896 |                 distance_m: 25.0 + (t * 0.07).sin() * 10.0, | support logic; impact depends on neighboring bloc
| 897 |                 lateral_offset_m: 3.5, | support logic; impact depends on neighboring block and data flow. |
| 898 |                 speed_ms: 8.0, | support logic; impact depends on neighboring block and data flow. |
| 899 |                 rcs_dbsm: 8.0, | support logic; impact depends on neighboring block and data flow. |
| 900 |                 object_type: RadarObjectClass::Car, | support logic; impact depends on neighboring block and da
| 901 |             }, | support logic; impact depends on neighboring block and data flow. |
| 902 |             SimTrafficObj { | support logic; impact depends on neighboring block and data flow. |
| 903 |                 id: 2, | support logic; impact depends on neighboring block and data flow. |
| 904 |                 distance_m: 80.0, | support logic; impact depends on neighboring block and data flow. |
| 905 |                 lateral_offset_m: -0.5, | support logic; impact depends on neighboring block and data flow. |
| 906 |                 speed_ms: 15.0, | support logic; impact depends on neighboring block and data flow. |
| 907 |                 rcs_dbsm: 22.0, | support logic; impact depends on neighboring block and data flow. |
| 908 |                 object_type: RadarObjectClass::Truck, | support logic; impact depends on neighboring block and
| 909 |             }, | support logic; impact depends on neighboring block and data flow. |
| 910 |         ] | support logic; impact depends on neighboring block and data flow. |
| 911 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 912 | (blank) | blank separator; no runtime behavior. |
| 913 |     fn build_lidar_world(&self) -> Vec<WorldObstacle> { | function signature/body; changes affect API behavior a
| 914 |         let t = self.elapsed; | support logic; impact depends on neighboring block and data flow. |
| 915 |         vec![ | support logic; impact depends on neighboring block and data flow. |
| 916 |             WorldObstacle::new_vehicle(50.0 + (t * 0.1).sin() * 20.0, 0.0, 10.0), | support logic; impact depend
| 917 |             WorldObstacle::new_vehicle(25.0, 3.5, 8.0), | support logic; impact depends on neighboring block and
| 918 |             WorldObstacle::new_barrier(30.0, -8.0, 10.0), | support logic; impact depends on neighboring block a
| 919 |             WorldObstacle::new_pedestrian(12.0, 2.0), | support logic; impact depends on neighboring block and
| 920 |         ] | support logic; impact depends on neighboring block and data flow. |
| 921 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 922 | (blank) | blank separator; no runtime behavior. |
| 923 |     // ═══════════════════════════════════════════════════════════════════════════════ | comment/d
| 924 |     fn apply_fault(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 925 |         match self.selected_fault { | branch dispatcher; arm changes alter behavior class handling. |
| 926 |             FaultType::HighCoolantTemp => { | support logic; impact depends on neighboring block and data flow.
| 927 |                 self.ecm.coolant_temp_c = 115.0; | support logic; impact depends on neighboring block and data f
| 928 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 929 |             FaultType::LowOilPressure => { | support logic; impact depends on neighboring block and data flow.
| 930 |                 self.ecm.oil_pressure_kpa = 40.0; | support logic; impact depends on neighboring block and data
| 931 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 932 |             FaultType::LowFuelPressure => { | support logic; impact depends on neighboring block and data flow.
| 933 |                 self.ecm.fuel_pressure_kpa = 50.0; | support logic; impact depends on neighboring block and data
| 934 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 935 |             FaultType::CriticalDefLevel => { | support logic; impact depends on neighboring block and data flow
| 936 |                 self.ecm.def_level_pct = 1.0; | support logic; impact depends on neighboring block and data flo
| 937 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 938 |             FaultType::HighDpfSoot => { | support logic; impact depends on neighboring block and data flow. |
| 939 |                 self.ecm.dpf_soot_pct = 95.0; | support logic; impact depends on neighboring block and data flo
| 940 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 941 |             FaultType::HighBoostPressure => { | support logic; impact depends on neighboring block and data flo
| 942 |                 self.ecm.boost_pressure_kpa = 340.0; | support logic; impact depends on neighboring block and da
| 943 |                 self.ecm.intake_manifold_kpa = 340.0; | support logic; impact depends on neighboring block and
```

```
| 944 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 945 |             FaultType::LowBoostPressure => { | support logic; impact depends on neighboring block and data flow
| 946 |                 self.ecm.boost_pressure_kpa = 80.0; | support logic; impact depends on neighboring block and dat
| 947 |                 self.ecm.intake_manifold_kpa = 82.0; | support logic; impact depends on neighboring block and da
| 948 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 949 |             FaultType::LowCoolantPressure => { | support logic; impact depends on neighboring block and data flo
| 950 |                 self.ecm.coolant_pres_kpa = 40.0; | support logic; impact depends on neighboring block and data
| 951 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 952 |             FaultType::HighIntakeTemp => { | support logic; impact depends on neighboring block and data flow.
| 953 |                 self.ecm.intake_temp_c = 115.0; | support logic; impact depends on neighboring block and data fl
| 954 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 955 |             FaultType::HighExhaustTemp => { | support logic; impact depends on neighboring block and data flow.
| 956 |                 self.ecm.exhaust_temp_c = 860.0; | support logic; impact depends on neighboring block and data f
| 957 |                 self.ecm.dpf_temp_c = 780.0; | support logic; impact depends on neighboring block and data flow
| 958 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 959 |             FaultType::PoorDefQuality => { | support logic; impact depends on neighboring block and data flow.
| 960 |                 self.ecm.def_quality_pct = 55.0; | support logic; impact depends on neighboring block and data f
| 961 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 962 |             FaultType::HighNoxTailpipe => { | support logic; impact depends on neighboring block and data flow.
| 963 |                 self.ecm.nox_tailpipe_ppm = 1100.0; | support logic; impact depends on neighboring block and dat
| 964 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 965 |             FaultType::LowFuelRailPressure => { | support logic; impact depends on neighboring block and data fl
| 966 |                 self.ecm.fuel_rail_pressure_mpa = 45.0; | support logic; impact depends on neighboring block and
| 967 |                 self.ecm.fuel_pressure_kpa = 120.0; | support logic; impact depends on neighboring block and dat
| 968 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 969 |             FaultType::TransClutchOverheat => { | support logic; impact depends on neighboring block and data fl
| 970 |                 self.tcm.clutch_temp_c = 250.0; | support logic; impact depends on neighboring block and data fl
| 971 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 972 |             FaultType::TransNeutralFailure => { /* TCM cannot leave neutral */ } | support logic; impact depends
| 973 |             FaultType::LowBatteryVoltage => { | support logic; impact depends on neighboring block and data flow
| 974 |                 self.bcm.battery_voltage = 10.8; | support logic; impact depends on neighboring block and data f
| 975 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 976 |             FaultType::AlternatorFault => { | support logic; impact depends on neighboring block and data flow.
| 977 |                 self.ecm.alternator_v = 11.0; | support logic; impact depends on neighboring block and data flow
| 978 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 979 |             FaultType::HighHydTemp => { | support logic; impact depends on neighboring block and data flow. |
| 980 |                 self.hcm.fluid_temp_c = 115.0; | support logic; impact depends on neighboring block and data flo
| 981 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 982 |             FaultType::LowHydLevel => { | support logic; impact depends on neighboring block and data flow. |
| 983 |                 self.hcm.fluid_level_pct = 10.0; | support logic; impact depends on neighboring block and data f
| 984 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 985 |             FaultType::HydFilterBypass => { | support logic; impact depends on neighboring block and data flow.
| 986 |                 self.hcm.filter_dp_bar = 8.0; | support logic; impact depends on neighboring block and data flow
| 987 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 988 |             FaultType::BusOffInjection => { | support logic; impact depends on neighboring block and data flow.
| 989 |                 self.gateway.inject_bus_off = true; | support logic; impact depends on neighboring block and dat
| 990 |                 self.fault_active = false; | support logic; impact depends on neighboring block and data flow.
| 991 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 992 |             FaultType::KillEcm => { | support logic; impact depends on neighboring block and data flow. |
| 993 |                 self.gateway.kill_node(j1939::addr::ECM_1); | support logic; impact depends on neighboring bloc
| 994 |                 self.fault_active = false; | support logic; impact depends on neighboring block and data flow.
| 995 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 996 |             FaultType::KillTcm => { | support logic; impact depends on neighboring block and data flow. |
| 997 |                 self.gateway.kill_node(j1939::addr::TRANSMISSION); | support logic; impact depends on neighborin
| 998 |                 self.fault_active = false; | support logic; impact depends on neighboring block and data flow.
| 999 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1000 |              FaultType::None => {} | support logic; impact depends on neighboring block and data flow. |
| 1001 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1002 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1003 | (blank) | blank separator; no runtime behavior. |
| 1004 |     // ■■■
```

```
| 1020 |     /// Advance ignition key one position | comment/documentation; changing affects understanding and intent tr
| 1021 |     pub fn key_advance(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1022 |         self.boot.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 1023 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1024 | (blank) | blank separator; no runtime behavior. |
| 1025 |     /// Turn key fully off — powers down all ECUs | comment/documentation; changing affects understanding and i
| 1026 |     pub fn key_off(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1027 |         self.boot.key_off(); | support logic; impact depends on neighboring block and data flow. |
| 1028 |         self.throttle_pct = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1029 |         self.brake_pct = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1030 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1031 | (blank) | blank separator; no runtime behavior. |
| 1032 |     /// Hard reset — rebuilds entire bench from scratch | comment/documentation; changing affects understanding
| 1033 |     pub fn reset(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1034 |         *self = Self::new(); | support logic; impact depends on neighboring block and data flow. |
| 1035 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1036 | (blank) | blank separator; no runtime behavior. |
| 1037 |     pub fn apply_leak_manual_params( | function signature/body; changes affect API behavior and control-flow.
| 1038 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 1039 |         circuit_name: &str, | support logic; impact depends on neighboring block and data flow. |
| 1040 |         params: ManualCircuitParams, | support logic; impact depends on neighboring block and data flow. |
| 1041 |     ) -> bool { | support logic; impact depends on neighboring block and data flow. |
| 1042 |         self.leak_rig.apply_manual_params(circuit_name, params) | support logic; impact depends on neighboring
| 1043 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1044 | (blank) | blank separator; no runtime behavior. |
| 1045 |     pub fn add_custom_leak_circuit(&mut self, circuit: LeakCircuit) { | function signature/body; changes affect
| 1046 |         self.leak_rig.add_circuit(circuit); | support logic; impact depends on neighboring block and data flow
| 1047 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1048 | (blank) | blank separator; no runtime behavior. |
| 1049 |     pub fn predict_leak_scenarios(&self, horizon_s: f64, dt: f64) -> Vec<ScenarioPrediction> { | function signa
| 1050 |         self.leak_rig.predict_scenarios(horizon_s, dt) | support logic; impact depends on neighboring block and
| 1051 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1052 | (blank) | blank separator; no runtime behavior. |
| 1053 |     pub fn export_leak_report_json<P: AsRef<std::path::Path>>( | function signature/body; changes affect API be
| 1054 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1055 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1056 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1057 |         self.leak_rig.export_last_results_json(path) | support logic; impact depends on neighboring block and
| 1058 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1059 | (blank) | blank separator; no runtime behavior. |
| 1060 |     pub fn export_leak_report_csv<P: AsRef<std::path::Path>>( | function signature/body; changes affect API bel
| 1061 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1062 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1063 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1064 |         self.leak_rig.export_last_results_csv(path) | support logic; impact depends on neighboring block and da
| 1065 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1066 | (blank) | blank separator; no runtime behavior. |
| 1067 |     pub fn export_leak_predictions_json<P: AsRef<std::path::Path>>( | function signature/body; changes affect A
| 1068 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1069 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1070 |         predictions: &[ScenarioPrediction], | support logic; impact depends on neighboring block and data flow
| 1071 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1072 |         self.leak_rig.export_predictions_json(path, predictions) | support logic; impact depends on neighboring
| 1073 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1074 | (blank) | blank separator; no runtime behavior. |
| 1075 |     pub fn export_leak_predictions_csv<P: AsRef<std::path::Path>>( | function signature/body; changes affect AP
| 1076 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1077 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1078 |         predictions: &[ScenarioPrediction], | support logic; impact depends on neighboring block and data flow
| 1079 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1080 |         self.leak_rig.export_predictions_csv(path, predictions) | support logic; impact depends on neighboring
| 1081 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1082 | (blank) | blank separator; no runtime behavior. |
| 1083 |     pub fn export_leak_material_catalog_json<P: AsRef<std::path::Path>>( | function signature/body; changes af
| 1084 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1085 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1086 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1087 |         self.leak_rig.export_material_catalog_json(path) | support logic; impact depends on neighboring block a
| 1088 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1089 | (blank) | blank separator; no runtime behavior. |
| 1090 |     pub fn export_leak_material_catalog_csv<P: AsRef<std::path::Path>>( | function signature/body; changes affe
| 1091 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1092 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1093 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1094 |         self.leak_rig.export_material_catalog_csv(path) | support logic; impact depends on neighboring block a
| 1095 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 1096 | (blank) | blank separator; no runtime behavior. |
| 1097 |     pub fn export_leak_oil_catalog_json<P: AsRef<std::path::Path>>( | function signature/body; changes affect A
| 1098 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1099 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1100 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1101 |         self.leak_rig.export_oil_catalog_json(path) | support logic; impact depends on neighboring block and da
| 1102 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1103 | (blank) | blank separator; no runtime behavior. |
| 1104 |     pub fn export_leak_oil_catalog_csv<P: AsRef<std::path::Path>>( | function signature/body; changes affect AP
| 1105 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1106 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1107 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1108 |         self.leak_rig.export_oil_catalog_csv(path) | support logic; impact depends on neighboring block and dat
| 1109 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1110 | (blank) | blank separator; no runtime behavior. |
| 1111 |     pub fn calibrate_leak_model_from_csv<P: AsRef<std::path::Path>>( | function signature/body; changes affect
| 1112 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 1113 |         csv_path: P, | support logic; impact depends on neighboring block and data flow. |
| 1114 |     ) -> std::io::Result<CalibrationReport> { | support logic; impact depends on neighboring block and data flo
| 1115 |         self.leak_rig.calibrate_from_csv(csv_path) | support logic; impact depends on neighboring block and dat
| 1116 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1117 | (blank) | blank separator; no runtime behavior. |
| 1118 |     pub fn export_leak_calibration_report_json<P: AsRef<std::path::Path>>( | function signature/body; changes a
| 1119 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1120 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1121 |         report: &CalibrationReport, | support logic; impact depends on neighboring block and data flow. |
| 1122 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1123 |         self.leak_rig.export_calibration_report_json(path, report) | support logic; impact depends on neighbori
| 1124 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1125 | (blank) | blank separator; no runtime behavior. |
| 1126 |     pub fn export_leak_calibration_report_csv<P: AsRef<std::path::Path>>( | function signature/body; changes af
| 1127 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1128 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1129 |         report: &CalibrationReport, | support logic; impact depends on neighboring block and data flow. |
| 1130 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1131 |         self.leak_rig.export_calibration_report_csv(path, report) | support logic; impact depends on neighborin
| 1132 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1133 | (blank) | blank separator; no runtime behavior. |
| 1134 |     pub fn monte_carlo_leak_predictions( | function signature/body; changes affect API behavior and control-flo
| 1135 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1136 |         runs: usize, | support logic; impact depends on neighboring block and data flow. |
| 1137 |         horizon_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 1138 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 1139 |         variation_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 1140 |     ) -> Vec<ScenarioPrediction> { | support logic; impact depends on neighboring block and data flow. |
| 1141 |         self.leak_rig | support logic; impact depends on neighboring block and data flow. |
| 1142 |             .run_monte_carlo(runs, horizon_s, dt, variation_pct) | support logic; impact depends on neighboring
| 1143 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1144 | (blank) | blank separator; no runtime behavior. |
| 1145 |     pub fn export_can_snapshot_json<P: AsRef<std::path::Path>>( | function signature/body; changes affect API b
| 1146 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1147 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1148 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1149 |         self.can_net.export_snapshot_json(path) | support logic; impact depends on neighboring block and data f
| 1150 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1151 | (blank) | blank separator; no runtime behavior. |
| 1152 |     pub fn export_can_snapshot_csv<P: AsRef<std::path::Path>>( | function signature/body; changes affect API be
| 1153 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 1154 |         path: P, | support logic; impact depends on neighboring block and data flow. |
| 1155 |     ) -> std::io::Result<()> { | support logic; impact depends on neighboring block and data flow. |
| 1156 |         self.can_net.export_snapshot_csv(path) | support logic; impact depends on neighboring block and data fl
| 1157 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1158 | (blank) | blank separator; no runtime behavior. |
| 1159 |     // ■ Convenience accessors ■■■
```

# File Deep Dive: src/lidar.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 356 | Non-empty: 324 | Symbol count: 18

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| ray_intersect | 2 |
| new | 2 |
| clear_grid | 2 |
| push | 2 |
| len | 2 |
| cluster_obstacles | 2 |
| fmt | 1 |
| new_vehicle | 1 |
| new_pedestrian | 1 |
| new_barrier | 1 |
| new_vlp16 | 1 |
| update | 1 |
| cell_occupied | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 9 | struct | LidarPoint | data/schema coupling and match/serde break risk |
| 21 | struct | LidarCluster | data/schema coupling and match/serde break risk |
| 35 | enum | LidarObjectType | data/schema coupling and match/serde break risk |
| 44 | impl | std::fmt::Display for LidarObjectType | method behavior and trait semantics |
| 45 | fn | fmt | API and behavior coupling to callers/implementers |
| 58 | struct | WorldObstacle | data/schema coupling and match/serde break risk |
| 68 | impl | WorldObstacle | method behavior and trait semantics |
| 69 | fn | new_vehicle | API and behavior coupling to callers/implementers |
| 80 | fn | new_pedestrian | API and behavior coupling to callers/implementers |
| 91 | fn | new_barrier | API and behavior coupling to callers/implementers |
| 104 | fn | ray_intersect | API and behavior coupling to callers/implementers |
| 141 | struct | LidarSensor | data/schema coupling and match/serde break risk |
| 175 | impl | LidarSensor | method behavior and trait semantics |
| 176 | fn | new_vlp16 | API and behavior coupling to callers/implementers |
| 201 | fn | update | API and behavior coupling to callers/implementers |
| 302 | fn | cluster_obstacles | API and behavior coupling to callers/implementers |
| 344 | fn | clear_grid | API and behavior coupling to callers/implementers |
| 353 | fn | cell_occupied | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! LIDAR — 3D rotating LIDAR simulation (Velodyne VLP-16 / Ouster OS1-64 style). | comment/documentation; changin
| 2 | //! Generates real point clouds by raycasting against simulated environment obstacles. | comment/documentation; ch
| 3 | //! 16 vertical channels, 360° horizontal, 10 Hz rotation rate. | comment/documentation; changing affects understa
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use std::f64::consts::PI; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | // ■■ Point ■■■
```

```
| 24 |     pub centroid_y: f64, | support logic; impact depends on neighboring block and data flow. |
| 25 |     pub centroid_z: f64, | support logic; impact depends on neighboring block and data flow. |
| 26 |     pub length_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 27 |     pub width_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 28 |     pub height_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 29 |     pub distance_m: f64, // front-to-front distance | support logic; impact depends on neighboring block and data
| 30 |     pub object_type: LidarObjectType, | support logic; impact depends on neighboring block and data flow. |
| 31 |     pub point_count: usize, | support logic; impact depends on neighboring block and data flow. |
| 32 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 35 | pub enum LidarObjectType { | state machine variants; changes ripple through matches and business logic. |
| 36 |     Vehicle, | support logic; impact depends on neighboring block and data flow. |
| 37 |     Pedestrian, | support logic; impact depends on neighboring block and data flow. |
| 38 |     Cyclist, | support logic; impact depends on neighboring block and data flow. |
| 39 |     Barrier, | support logic; impact depends on neighboring block and data flow. |
| 40 |     GroundPoint, | support logic; impact depends on neighboring block and data flow. |
| 41 |     Unknown, | support logic; impact depends on neighboring block and data flow. |
| 42 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 43 | (blank) | blank separator; no runtime behavior. |
| 44 | impl std::fmt::Display for LidarObjectType { | implementation binding; behavior semantics and trait conformance
| 45 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 46 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 47 |             LidarObjectType::Vehicle => write!(f, "Vehicle   "), | protocol or I/O critical; changes may impact
| 48 |             LidarObjectType::Pedestrian => write!(f, "Pedestrian"), | protocol or I/O critical; changes may impa
| 49 |             LidarObjectType::Cyclist => write!(f, "Cyclist   "), | protocol or I/O critical; changes may impact
| 50 |             LidarObjectType::Barrier => write!(f, "Barrier   "), | protocol or I/O critical; changes may impact
| 51 |             LidarObjectType::GroundPoint => write!(f, "Ground    "), | protocol or I/O critical; changes may imp
| 52 |             LidarObjectType::Unknown => write!(f, "Unknown   "), | protocol or I/O critical; changes may impact
| 53 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 56 | (blank) | blank separator; no runtime behavior. |
| 57 | // ■■ World obstacle (for ray casting) ■■■
```

```
| 100 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 101 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 102 | (blank) | blank separator; no runtime behavior. |
| 103 |     /// Returns distance to ray intersection, or None if no hit | comment/documentation; changing affects unders
| 104 |     pub fn ray_intersect( | function signature/body; changes affect API behavior and control-flow. |
| 105 |         &self, | support logic; impact depends on neighboring block and data flow. |
| 106 |         origin_x: f64, | support logic; impact depends on neighboring block and data flow. |
| 107 |         origin_y: f64, | support logic; impact depends on neighboring block and data flow. |
| 108 |         dir_x: f64, | support logic; impact depends on neighboring block and data flow. |
| 109 |         dir_y: f64, | support logic; impact depends on neighboring block and data flow. |
| 110 |         z: f64, | support logic; impact depends on neighboring block and data flow. |
| 111 |     ) -> Option<f64> { | support logic; impact depends on neighboring block and data flow. |
| 112 |         if z < self.z_min - 0.1 \|\| z > self.z_max + 0.1 { | runtime gate; condition changes can enable/disabl
| 113 |             return None; | support logic; impact depends on neighboring block and data flow. |
| 114 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 115 |         // AABB intersection | comment/documentation; changing affects understanding and intent traceability. |
| 116 |         let inv_dx = if dir_x.abs() > 1e-10 { | support logic; impact depends on neighboring block and data flo
| 117 |             1.0 / dir_x | support logic; impact depends on neighboring block and data flow. |
| 118 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 119 |             1e10 | support logic; impact depends on neighboring block and data flow. |
| 120 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 121 |         let inv_dy = if dir_y.abs() > 1e-10 { | support logic; impact depends on neighboring block and data flo
| 122 |             1.0 / dir_y | support logic; impact depends on neighboring block and data flow. |
| 123 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 124 |             1e10 | support logic; impact depends on neighboring block and data flow. |
| 125 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 126 |         let t1x = ((self.x - self.half_l) - origin_x) * inv_dx; | support logic; impact depends on neighboring b
| 127 |         let t2x = ((self.x + self.half_l) - origin_x) * inv_dx; | support logic; impact depends on neighboring b
| 128 |         let t1y = ((self.y - self.half_w) - origin_y) * inv_dy; | support logic; impact depends on neighboring b
| 129 |         let t2y = ((self.y + self.half_w) - origin_y) * inv_dy; | support logic; impact depends on neighboring b
| 130 |         let tmin = t1x.min(t2x).max(t1y.min(t2y)); | support logic; impact depends on neighboring block and data
| 131 |         let tmax = t1x.max(t2x).min(t1y.max(t2y)); | support logic; impact depends on neighboring block and data
| 132 |         if tmax >= tmin && tmin > 0.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 133 |             Some(tmin) | support logic; impact depends on neighboring block and data flow. |
| 134 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 135 |             None | support logic; impact depends on neighboring block and data flow. |
| 136 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 137 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 139 | (blank) | blank separator; no runtime behavior. |
| 140 | // ■■ LIDAR sensor ■■■
```

```
| 176 |     pub fn new_vlp16() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 177 |         LidarSensor { | support logic; impact depends on neighboring block and data flow. |
| 178 |             channels: 16, | support logic; impact depends on neighboring block and data flow. |
| 179 |             rotation_hz: 10.0, | support logic; impact depends on neighboring block and data flow. |
| 180 |             h_resolution: 0.2, | support logic; impact depends on neighboring block and data flow. |
| 181 |             max_range_m: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 182 |             min_range_m: 0.3, | support logic; impact depends on neighboring block and data flow. |
| 183 |             active: true, | support logic; impact depends on neighboring block and data flow. |
| 184 |             points: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 185 |             clusters: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 186 |             points_per_scan: 0, | support logic; impact depends on neighboring block and data flow. |
| 187 |             scan_time_ms: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 188 |             occupancy_grid: vec![vec![0u8; 100]; 100], | support logic; impact depends on neighboring block and
| 189 |             grid_size: 100, | support logic; impact depends on neighboring block and data flow. |
| 190 |             total_scans: 0, | support logic; impact depends on neighboring block and data flow. |
| 191 |             obstacles_seen: 0, | support logic; impact depends on neighboring block and data flow. |
| 192 |             ground_level: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 193 |             rotation_angle: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 194 |             scan_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 195 |             cluster_id_counter: 0, | support logic; impact depends on neighboring block and data flow. |
| 196 |             noise_t: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 197 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 198 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 199 | (blank) | blank separator; no runtime behavior. |
| 200 |     /// Update LIDAR — generates point cloud from world obstacles. | comment/documentation; changing affects un
| 201 |     pub fn update(&mut self, world: &[WorldObstacle], vehicle_pitch: f64, dt: f64) { | function signature/body;
| 202 |         self.noise_t += dt; | support logic; impact depends on neighboring block and data flow. |
| 203 |         self.scan_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 204 |         if self.scan_timer < 1.0 / self.rotation_hz { | runtime gate; condition changes can enable/disable crit
| 205 |             return; | support logic; impact depends on neighboring block and data flow. |
| 206 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 207 |         self.scan_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 208 | (blank) | blank separator; no runtime behavior. |
| 209 |         let t0 = std::time::Instant::now(); | support logic; impact depends on neighboring block and data flow.
| 210 |         self.points.clear(); | support logic; impact depends on neighboring block and data flow. |
| 211 |         self.clear_grid(); | support logic; impact depends on neighboring block and data flow. |
| 212 | (blank) | blank separator; no runtime behavior. |
| 213 |         // VLP-16 vertical angles: -15° to +15° in 2° steps | comment/documentation; changing affects understan
| 214 |         let vert_angles: Vec<f64> = (0..self.channels as i32) | support logic; impact depends on neighboring bl
| 215 |             .map(\|i\| -15.0 + i as f64 * 2.0) // degrees | support logic; impact depends on neighboring block
| 216 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 217 | (blank) | blank separator; no runtime behavior. |
| 218 |         let n = \|s: f64\| ((s * 127.1 + 311.7).sin() * 43758.5 % 1.0 - 0.5) * 2.0; | support logic; impact dep
| 219 |         let nt = self.noise_t; | support logic; impact depends on neighboring block and data flow. |
| 220 | (blank) | blank separator; no runtime behavior. |
| 221 |         // Full 360° scan | comment/documentation; changing affects understanding and intent traceability. |
| 222 |         let mut az = 0.0_f64; | support logic; impact depends on neighboring block and data flow. |
| 223 |         while az < 360.0 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 224 |             let az_rad = az * PI / 180.0; | support logic; impact depends on neighboring block and data flow. |
| 225 |             let dir_x = az_rad.cos(); | support logic; impact depends on neighboring block and data flow. |
| 226 |             let dir_y = az_rad.sin(); | support logic; impact depends on neighboring block and data flow. |
| 227 | (blank) | blank separator; no runtime behavior. |
| 228 |             for (ring, vert_deg) in vert_angles.iter().enumerate() { | iteration logic; may alter timing, throu
| 229 |                 let vert_rad = (vert_deg + vehicle_pitch) * PI / 180.0; | support logic; impact depends on neig
| 230 |                 let h_factor = vert_rad.cos(); | support logic; impact depends on neighboring block and data fl
| 231 |                 let z_per_m = vert_rad.sin(); | support logic; impact depends on neighboring block and data flo
| 232 | (blank) | blank separator; no runtime behavior. |
| 233 |                 // Find closest intersection | comment/documentation; changing affects understanding and intent
| 234 |                 let mut min_dist = self.max_range_m; | support logic; impact depends on neighboring block and d
| 235 |                 let mut hit_intensity = 0u8; | support logic; impact depends on neighboring block and data flow
| 236 | (blank) | blank separator; no runtime behavior. |
| 237 |                 // Ground return | comment/documentation; changing affects understanding and intent traceabilit
| 238 |                 if *vert_deg < 0.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 239 |                     let ground_dist = (self.ground_level - 1.8) / (-vert_rad.sin()).max(1e-6); | support logic;
| 240 |                     if ground_dist < min_dist && ground_dist > self.min_range_m { | runtime gate; condition cha
| 241 |                         min_dist = ground_dist; | support logic; impact depends on neighboring block and data f
| 242 |                         hit_intensity = 30; | support logic; impact depends on neighboring block and data flow.
| 243 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 244 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 245 | (blank) | blank separator; no runtime behavior. |
| 246 |                 // Obstacle returns | comment/documentation; changing affects understanding and intent traceabi
| 247 |                 for obs in world { | iteration logic; may alter timing, throughput, and safety constraints. |
| 248 |                     let z_at_range = \|r: f64\| self.ground_level + z_per_m * r + 1.8; // sensor height 1.8m |
| 249 |                                                                                         // Check at expected obsta
| 250 |                     if let Some(dist) = obs.ray_intersect( | runtime gate; condition changes can enable/disable
| 251 |                         0.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 252 |                         0.0, | support logic; impact depends on neighboring block and data flow. |
| 253 |                         dir_x * h_factor, | support logic; impact depends on neighboring block and data flow. |
| 254 |                         dir_y * h_factor, | support logic; impact depends on neighboring block and data flow. |
| 255 |                         z_at_range(obs.x.hypot(obs.y)), | support logic; impact depends on neighboring block and
| 256 |                     ) { | support logic; impact depends on neighboring block and data flow. |
| 257 |                         if dist < min_dist { | runtime gate; condition changes can enable/disable critical paths
| 258 |                             min_dist = dist; | support logic; impact depends on neighboring block and data flow
| 259 |                             let snr = obs.reflectivity * (1.0 - dist / self.max_range_m); | support logic; impac
| 260 |                             hit_intensity = (snr * 200.0 + 20.0).clamp(0.0, 255.0) as u8; | support logic; impac
| 261 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 262 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 263 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 264 | (blank) | blank separator; no runtime behavior. |
| 265 |                 if min_dist < self.max_range_m && min_dist > self.min_range_m { | runtime gate; condition change
| 266 |                     let noise_m = n(nt + az * 0.01 + ring as f64 * 0.1) * 0.02; // ±2cm noise | support logic;
| 267 |                     let d = (min_dist + noise_m) as f32; | support logic; impact depends on neighboring block a
| 268 |                     let px = d * dir_x as f32 * vert_rad.cos() as f32; | support logic; impact depends on neigh
| 269 |                     let py = d * dir_y as f32 * vert_rad.cos() as f32; | support logic; impact depends on neigh
| 270 |                     let pz = (self.ground_level + z_per_m * min_dist) as f32; | support logic; impact depends o
| 271 | (blank) | blank separator; no runtime behavior. |
| 272 |                     self.points.push(LidarPoint { | support logic; impact depends on neighboring block and data
| 273 |                         x: px, | support logic; impact depends on neighboring block and data flow. |
| 274 |                         y: py, | support logic; impact depends on neighboring block and data flow. |
| 275 |                         z: pz, | support logic; impact depends on neighboring block and data flow. |
| 276 |                         intensity: hit_intensity, | support logic; impact depends on neighboring block and data
| 277 |                         ring: ring as u8, | support logic; impact depends on neighboring block and data flow. |
| 278 |                         azimuth: az as f32, | support logic; impact depends on neighboring block and data flow.
| 279 |                         distance: d, | support logic; impact depends on neighboring block and data flow. |
| 280 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 281 | (blank) | blank separator; no runtime behavior. |
| 282 |                     // Update occupancy grid | comment/documentation; changing affects understanding and intent
| 283 |                     let gx = (50.0 + px as f64) as usize; | support logic; impact depends on neighboring block a
| 284 |                     let gy = (50.0 + py as f64) as usize; | support logic; impact depends on neighboring block a
| 285 |                     if gx < self.grid_size && gy < self.grid_size && pz > 0.2 { | runtime gate; condition change
| 286 |                         self.occupancy_grid[gy][gx] = | support logic; impact depends on neighboring block and
| 287 |                             self.occupancy_grid[gy][gx].saturating_add(10); | support logic; impact depends on
| 288 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 289 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 290 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 291 |             az += self.h_resolution; | support logic; impact depends on neighboring block and data flow. |
| 292 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 293 | (blank) | blank separator; no runtime behavior. |
| 294 |         self.points_per_scan = self.points.len() as u32; | support logic; impact depends on neighboring block a
| 295 |         self.scan_time_ms = t0.elapsed().as_secs_f64() * 1000.0; | support logic; impact depends on neighboring
| 296 |         self.total_scans += 1; | support logic; impact depends on neighboring block and data flow. |
| 297 | (blank) | blank separator; no runtime behavior. |
| 298 |         // Simple clustering: group nearby occupied grid cells | comment/documentation; changing affects unders
| 299 |         self.cluster_obstacles(world); | support logic; impact depends on neighboring block and data flow. |
| 300 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 301 | (blank) | blank separator; no runtime behavior. |
| 302 |     fn cluster_obstacles(&mut self, world: &[WorldObstacle]) { | function signature/body; changes affect API be
| 303 |         self.clusters.clear(); | support logic; impact depends on neighboring block and data flow. |
| 304 |         for obs in world.iter() { | iteration logic; may alter timing, throughput, and safety constraints. |
| 305 |             let dist = (obs.x.powi(2) + obs.y.powi(2)).sqrt(); | support logic; impact depends on neighboring b
| 306 |             if dist > self.max_range_m { | runtime gate; condition changes can enable/disable critical paths. |
| 307 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 308 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 309 |             let obj_type = if obs.z_max > 1.2 && obs.half_w > 0.5 { | support logic; impact depends on neighbor
| 310 |                 LidarObjectType::Vehicle | support logic; impact depends on neighboring block and data flow. |
| 311 |             } else if obs.z_max > 1.5 { | support logic; impact depends on neighboring block and data flow. |
| 312 |                 LidarObjectType::Pedestrian | support logic; impact depends on neighboring block and data flow.
| 313 |             } else if obs.z_max < 0.9 { | support logic; impact depends on neighboring block and data flow. |
| 314 |                 LidarObjectType::Barrier | support logic; impact depends on neighboring block and data flow. |
| 315 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 316 |                 LidarObjectType::Unknown | support logic; impact depends on neighboring block and data flow. |
| 317 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 318 |             let pts = self | support logic; impact depends on neighboring block and data flow. |
| 319 |                 .points | support logic; impact depends on neighboring block and data flow. |
| 320 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 321 |                 .filter(\|p\| { | support logic; impact depends on neighboring block and data flow. |
| 322 |                     ((p.x as f64 - obs.x).powi(2) + (p.y as f64 - obs.y).powi(2)).sqrt() < 3.0 | support logic;
| 323 |                 }) | support logic; impact depends on neighboring block and data flow. |
| 324 |                 .count(); | support logic; impact depends on neighboring block and data flow. |
| 325 |             if pts > 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 326 |                 self.clusters.push(LidarCluster { | support logic; impact depends on neighboring block and data
| 327 |                     id: self.cluster_id_counter, | support logic; impact depends on neighboring block and data
```

```
| 328 |                     centroid_x: obs.x, | support logic; impact depends on neighboring block and data flow. |
| 329 |                     centroid_y: obs.y, | support logic; impact depends on neighboring block and data flow. |
| 330 |                     centroid_z: obs.z_max / 2.0, | support logic; impact depends on neighboring block and data f
| 331 |                     length_m: obs.half_l * 2.0, | support logic; impact depends on neighboring block and data fl
| 332 |                     width_m: obs.half_w * 2.0, | support logic; impact depends on neighboring block and data flo
| 333 |                     height_m: obs.z_max, | support logic; impact depends on neighboring block and data flow. |
| 334 |                     distance_m: dist, | support logic; impact depends on neighboring block and data flow. |
| 335 |                     object_type: obj_type, | support logic; impact depends on neighboring block and data flow.
| 336 |                     point_count: pts, | support logic; impact depends on neighboring block and data flow. |
| 337 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 338 |                 self.cluster_id_counter = self.cluster_id_counter.wrapping_add(1); | support logic; impact depen
| 339 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 341 |         self.obstacles_seen = self.clusters.len(); | support logic; impact depends on neighboring block and data
| 342 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 343 | (blank) | blank separator; no runtime behavior. |
| 344 |     fn clear_grid(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 345 |         for row in &mut self.occupancy_grid { | iteration logic; may alter timing, throughput, and safety constr
| 346 |             for cell in row.iter_mut() { | iteration logic; may alter timing, throughput, and safety constraints
| 347 |                 *cell = (*cell).saturating_sub(5); | support logic; impact depends on neighboring block and data
| 348 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 349 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 350 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 351 | (blank) | blank separator; no runtime behavior. |
| 352 |     /// Get occupancy as percentage for a given grid cell | comment/documentation; changing affects understandin
| 353 |     pub fn cell_occupied(&self, row: usize, col: usize) -> bool { | function signature/body; changes affect API
| 354 |         row < self.grid_size && col < self.grid_size && self.occupancy_grid[row][col] > 50 | support logic; impa
| 355 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 356 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/main.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 9466 | Non-empty: 9212 | Symbol count: 114

## Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| new | 577 |
| push | 151 |
| digital_readout | 87 |
| len | 50 |
| bar_gauge | 50 |
| is_empty | 34 |
| warning_lamp | 17 |
| pick_save_path | 14 |
| arc_gauge | 12 |
| get | 11 |
| uds_send_and_log | 11 |
| default | 10 |
| name | 10 |
| engine_running | 7 |
| electrical_node_online | 7 |
| all | 6 |
| can_bus_from_idx | 6 |
| parse_u32_filter | 6 |
| ignition | 5 |
| set_direction | 4 |
| apply_manual_to_selected_circuit | 4 |
| ema | 4 |
| is_online | 4 |
| pick_open_path | 3 |
| oil_type_from_idx | 3 |

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 8 | module | io | namespace and compile-time linkage |
| 9 | module | ecm_mock_provider | namespace and compile-time linkage |
| 10 | module | widgets | namespace and compile-time linkage |
| 37 | const | DT | namespace and compile-time linkage |

| 38 | const | PLOT_WINDOW | namespace and compile-time linkage |
| 39 | const | EVENT_MAX | namespace and compile-time linkage |
| 40 | type | TraceRow | data/schema coupling and match/serde break risk |
| 46 | enum | Cmd | data/schema coupling and match/serde break risk |
| 145 | enum | Tab | data/schema coupling and match/serde break risk |
| 167 | struct | LeakManualUi | data/schema coupling and match/serde break risk |
| 181 | impl | Default for LeakManualUi | method behavior and trait semantics |
| 182 | fn | default | API and behavior coupling to callers/implementers |
| 200 | struct | LeakCustomUi | data/schema coupling and match/serde break risk |
| 226 | impl | Default for LeakCustomUi | method behavior and trait semantics |
| 227 | fn | default | API and behavior coupling to callers/implementers |
| 257 | enum | CanMode | data/schema coupling and match/serde break risk |
| 264 | enum | EventLevel | data/schema coupling and match/serde break risk |
| 273 | struct | AppEvent | data/schema coupling and match/serde break risk |
| 280 | struct | Signal | data/schema coupling and match/serde break risk |
| 294 | struct | App | data/schema coupling and match/serde break risk |
| 426 | impl | App | method behavior and trait semantics |
| 427 | fn | new | API and behavior coupling to callers/implementers |
| 542 | fn | pick_save_path | API and behavior coupling to callers/implementers |
| 553 | fn | pick_open_path | API and behavior coupling to callers/implementers |
| 560 | fn | refresh_mock_ecm_live_from_sim | API and behavior coupling to callers/implementers |
| 572 | fn | ecm_treeview | API and behavior coupling to callers/implementers |
| 603 | fn | oil_type_from_idx | API and behavior coupling to callers/implementers |
| 607 | fn | component_from_idx | API and behavior coupling to callers/implementers |
| 615 | fn | material_from_idx | API and behavior coupling to callers/implementers |
| 622 | fn | idx_from_oil_type | API and behavior coupling to callers/implementers |
| 626 | fn | sync_leak_manual_from_selected | API and behavior coupling to callers/implementers |
| 642 | fn | sanitize_leak_manual | API and behavior coupling to callers/implementers |
| 667 | fn | sanitize_leak_custom | API and behavior coupling to callers/implementers |
| 690 | fn | validate_leak_manual | API and behavior coupling to callers/implementers |
| 704 | fn | validate_leak_custom | API and behavior coupling to callers/implementers |
| 721 | fn | build_manual_leak_params | API and behavior coupling to callers/implementers |
| 738 | fn | apply_manual_to_selected_circuit | API and behavior coupling to callers/implementers |
| 754 | fn | can_bus_from_idx | API and behavior coupling to callers/implementers |
| 764 | fn | parse_u32_filter | API and behavior coupling to callers/implementers |
| 773 | fn | can_filter_match_signal | API and behavior coupling to callers/implementers |
| 799 | fn | can_filter_match_trace | API and behavior coupling to callers/implementers |
| 819 | fn | ema | API and behavior coupling to callers/implementers |
| 823 | fn | update_v2x_kpis | API and behavior coupling to callers/implementers |
| 829 | fn | approach_f32 | API and behavior coupling to callers/implementers |
| 839 | fn | shape_pedal | API and behavior coupling to callers/implementers |
| 846 | fn | flush_cmds | API and behavior coupling to callers/implementers |
| 1297 | fn | sync_params_from_bench | API and behavior coupling to callers/implementers |
| 1309 | fn | update_signals | API and behavior coupling to callers/implementers |
| 1377 | fn | detect_events | API and behavior coupling to callers/implementers |
| 1561 | fn | push_plots | API and behavior coupling to callers/implementers |
| 1581 | fn | auto_sequence | API and behavior coupling to callers/implementers |
| 1591 | fn | now_ms | API and behavior coupling to callers/implementers |
| 1601 | fn | main | API and behavior coupling to callers/implementers |
| 1651 | impl | eframe::App for App | method behavior and trait semantics |
| 1652 | fn | update | API and behavior coupling to callers/implementers |
| 1769 | impl | App | method behavior and trait semantics |
| 1770 | fn | tab_bar | API and behavior coupling to callers/implementers |
| 1847 | fn | ad_readiness_issues | API and behavior coupling to callers/implementers |
| 1867 | fn | ux_alerts | API and behavior coupling to callers/implementers |
| 1935 | fn | ux_context_items | API and behavior coupling to callers/implementers |
| 2204 | fn | render_ux_context_strip | API and behavior coupling to callers/implementers |
| 2222 | fn | statusbar | API and behavior coupling to callers/implementers |
| 2283 | fn | render_ux_guide | API and behavior coupling to callers/implementers |
| 2413 | fn | render_ux_quick_actions | API and behavior coupling to callers/implementers |
| 2560 | fn | electrical_wire_mm2 | API and behavior coupling to callers/implementers |
| 2580 | fn | electrical_cap_nf_per_m | API and behavior coupling to callers/implementers |
| 2594 | fn | electrical_node_online | API and behavior coupling to callers/implementers |
| 2601 | fn | tab_ecm_live_data | API and behavior coupling to callers/implementers |
| 2890 | fn | toolbar | API and behavior coupling to callers/implementers |
| 3116 | impl | App | method behavior and trait semantics |
| 3117 | fn | tab_help | API and behavior coupling to callers/implementers |
| 3199 | fn | tab_cluster | API and behavior coupling to callers/implementers |
| 3673 | impl | App | method behavior and trait semantics |
| 3674 | fn | tab_can | API and behavior coupling to callers/implementers |
| 3794 | fn | can_network_overview | API and behavior coupling to callers/implementers |
| 3819 | const | BUS_NAMES | namespace and compile-time linkage |
| 3995 | fn | can_signals | API and behavior coupling to callers/implementers |
| 4114 | fn | can_trace | API and behavior coupling to callers/implementers |
| 4284 | impl | App | method behavior and trait semantics |
| 4285 | fn | tab_events | API and behavior coupling to callers/implementers |

```
| 4437 | impl | App | method behavior and trait semantics |
| 4438 | fn | tab_ecu_net | API and behavior coupling to callers/implementers |
| 4621 | impl | App | method behavior and trait semantics |
| 4622 | fn | tab_engine | API and behavior coupling to callers/implementers |
| 4974 | impl | App | method behavior and trait semantics |
| 4975 | fn | dtc_fault_storage_addr | API and behavior coupling to callers/implementers |
| 4980 | fn | dtc_dm1_payload_hex | API and behavior coupling to callers/implementers |
| 4997 | fn | fault_selection_hint | API and behavior coupling to callers/implementers |
| 5018 | fn | tab_faults | API and behavior coupling to callers/implementers |
| 5327 | impl | App | method behavior and trait semantics |
| 5328 | fn | tab_boot | API and behavior coupling to callers/implementers |
| 5460 | impl | App | method behavior and trait semantics |
| 5461 | fn | tab_implements | API and behavior coupling to callers/implementers |
| 5918 | impl | App | method behavior and trait semantics |
| 5919 | fn | tab_params | API and behavior coupling to callers/implementers |
| 6235 | impl | App | method behavior and trait semantics |
| 6236 | fn | uds_process_sa | API and behavior coupling to callers/implementers |
| 6246 | fn | uds_send_and_log | API and behavior coupling to callers/implementers |
| 6278 | fn | run_uds_flash_pipeline | API and behavior coupling to callers/implementers |
| 6360 | fn | leak_ascii_cad | API and behavior coupling to callers/implementers |
| 6364 | const | SECTORS | namespace and compile-time linkage |
| 6468 | fn | tab_leak_lab | API and behavior coupling to callers/implementers |
| 7022 | const | COMP_NAMES | namespace and compile-time linkage |
| 7327 | fn | tab_electrical | API and behavior coupling to callers/implementers |
| 7676 | impl | App | method behavior and trait semantics |
| 7677 | fn | tab_sensors | API and behavior coupling to callers/implementers |
| 8067 | impl | App | method behavior and trait semantics |
| 8068 | fn | tab_autonomous | API and behavior coupling to callers/implementers |
| 8623 | impl | App | method behavior and trait semantics |
| 8624 | fn | tab_v2x | API and behavior coupling to callers/implementers |
| 9039 | impl | App | method behavior and trait semantics |
| 9040 | fn | tab_uds | API and behavior coupling to callers/implementers |
| 9357 | impl | App | method behavior and trait semantics |
| 9358 | fn | tab_plots | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Heavy Machinery ECU Bench — GUI v3.0 | comment/documentation; changing affects understanding and intent tracea
| 2 | //! Command-queue architecture: all UI→simulation mutations go through Cmd enum, | comment/documentation; changin
| 3 | //! executed AFTER the UI frame, eliminating all borrow-checker closure conflicts. | comment/documentation; chang
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | #![cfg_attr(not(debug_assertions), windows_subsystem = "windows")] | comment/documentation; changing affects unde
| 6 | #![allow(float_literal_f32_fallback)] | comment/documentation; changing affects understanding and intent traceabi
| 7 | (blank) | blank separator; no runtime behavior. |
| 8 | pub mod io; | module declaration; changing can break namespace graph. |
| 9 | mod ecm_mock_provider; | module declaration; changing can break namespace graph. |
| 10 | mod widgets; | module declaration; changing can break namespace graph. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | use auto_breaking::{ | import wiring; changing path/name can break compilation and module linkage. |
| 13 |     autonomous::FeatureState, | support logic; impact depends on neighboring block and data flow. |
| 14 |     boot_sequence::EcuBootStage, | support logic; impact depends on neighboring block and data flow. |
| 15 |     can_gateway::BusState, | support logic; impact depends on neighboring block and data flow. |
| 16 |     CalibrationReport, | support logic; impact depends on neighboring block and data flow. |
| 17 |     ElectricalControlPoint, ElectricalFaultMode, | support logic; impact depends on neighboring block and data f
| 18 |     ecu_abs::EspCondition, | support logic; impact depends on neighboring block and data flow. |
| 19 |     ecu_ecm::AftertreatmentState, | support logic; impact depends on neighboring block and data flow. |
| 20 |     ecu_tcm::{AutoShiftMode, ClutchState, Direction}, | support logic; impact depends on neighboring block and da
| 21 |     implement::PtoMode, | support logic; impact depends on neighboring block and data flow. |
| 22 |     j1939::DtcSeverity, | support logic; impact depends on neighboring block and data flow. |
| 23 |     v2x_telematics::ConnectState, | support logic; impact depends on neighboring block and data flow. |
| 24 |     CircuitComponent, HeavyMachinery, IgnitionState, LeakCircuit, ManualCircuitParams, OilType, | support logic;
| 25 |     OringMaterial, OringSpec, PressureEnvelope, ScenarioPrediction, FAULT_TYPES, | support logic; impact depends
| 26 | }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 27 | use chrono::Local; | import wiring; changing path/name can break compilation and module linkage. |
| 28 | use eframe::{egui, NativeOptions}; | import wiring; changing path/name can break compilation and module linkage.
| 29 | use egui::*; | import wiring; changing path/name can break compilation and module linkage. |
| 30 | use egui_plot::{Line, Plot, PlotPoints}; | import wiring; changing path/name can break compilation and module li
| 31 | use rfd::FileDialog; | import wiring; changing path/name can break compilation and module linkage. |
| 32 | use std::collections::{BTreeMap, HashMap, VecDeque}; | import wiring; changing path/name can break compilation a
| 33 | use std::path::PathBuf; | import wiring; changing path/name can break compilation and module linkage. |
| 34 | use std::time::Duration; | import wiring; changing path/name can break compilation and module linkage. |
| 35 | use widgets::{arc_gauge, bar_gauge, digital_readout, direction_selector, warning_lamp}; | import wiring; changing
| 36 | (blank) | blank separator; no runtime behavior. |
| 37 | const DT: f64 = 1.0 / 60.0; | support logic; impact depends on neighboring block and data flow. |
```

```
| 38 | const PLOT_WINDOW: usize = 600; | support logic; impact depends on neighboring block and data flow. |
| 39 | const EVENT_MAX: usize = 500; | support logic; impact depends on neighboring block and data flow. |
| 40 | type TraceRow = (f64, u32, u8, u8, String, String, String); | support logic; impact depends on neighboring block
| 41 | (blank) | blank separator; no runtime behavior. |
| 42 | // ═══════════════════════════════════════════════════════════════════════════════ | comment/d
| 43 | // Command queue — ALL mutations to bench go through here, no closures needed | comment/documentation; changing a
| 44 | // ═══════════════════════════════════════════════════════════════════════════════ | comment/d
| 45 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 46 | enum Cmd { | state machine variants; changes ripple through matches and business logic. |
| 47 |     // Ignition | comment/documentation; changing affects understanding and intent traceability. |
| 48 |     KeyAdvance, | support logic; impact depends on neighboring block and data flow. |
| 49 |     KeyOff, | support logic; impact depends on neighboring block and data flow. |
| 50 |     // Throttle/brake (raw value 0-100) | comment/documentation; changing affects understanding and intent tracea
| 51 |     SetThrottle(f32), | support logic; impact depends on neighboring block and data flow. |
| 52 |     SetBrake(f32), | support logic; impact depends on neighboring block and data flow. |
| 53 |     // Direction | comment/documentation; changing affects understanding and intent traceability. |
| 54 |     SetDirection(Direction), | support logic; impact depends on neighboring block and data flow. |
| 55 |     SetNeutral, | support logic; impact depends on neighboring block and data flow. |
| 56 |     // Gearbox | comment/documentation; changing affects understanding and intent traceability. |
| 57 |     ToggleAutoShift, | support logic; impact depends on neighboring block and data flow. |
| 58 |     ToggleCreeper, | support logic; impact depends on neighboring block and data flow. |
| 59 |     ManualUp, | support logic; impact depends on neighboring block and data flow. |
| 60 |     ManualDn, | support logic; impact depends on neighboring block and data flow. |
| 61 |     // ECM parameter overrides | comment/documentation; changing affects understanding and intent traceability.
| 62 |     SetFuelLevel(f64), | support logic; impact depends on neighboring block and data flow. |
| 63 |     SetDefLevel(f64), | support logic; impact depends on neighboring block and data flow. |
| 64 |     SetCoolantTemp(f64), | support logic; impact depends on neighboring block and data flow. |
| 65 |     SetOilPressure(f64), | support logic; impact depends on neighboring block and data flow. |
| 66 |     SetBoostPressure(f64), | support logic; impact depends on neighboring block and data flow. |
| 67 |     SetEngineHours(f64), | support logic; impact depends on neighboring block and data flow. |
| 68 |     ClearDtcs, | support logic; impact depends on neighboring block and data flow. |
| 69 |     // BCM | comment/documentation; changing affects understanding and intent traceability. |
| 70 |     ToggleWorkLights, | support logic; impact depends on neighboring block and data flow. |
| 71 |     ToggleRoadLights, | support logic; impact depends on neighboring block and data flow. |
| 72 |     ToggleBeacon, | support logic; impact depends on neighboring block and data flow. |
| 73 |     Honk, | support logic; impact depends on neighboring block and data flow. |
| 74 |     CycleWiper, | support logic; impact depends on neighboring block and data flow. |
| 75 |     ToggleHvac, | support logic; impact depends on neighboring block and data flow. |
| 76 |     ResetBcmFuses, | support logic; impact depends on neighboring block and data flow. |
| 77 |     ElectricalSetBranchFault(String, ElectricalFaultMode), | support logic; impact depends on neighboring block a
| 78 |     ElectricalSetControlFault(ElectricalControlPoint, ElectricalFaultMode), | support logic; impact depends on ne
| 79 |     ElectricalResetFaults, | support logic; impact depends on neighboring block and data flow. |
| 80 |     // Implements | comment/documentation; changing affects understanding and intent traceability. |
| 81 |     SetPtoMode(PtoMode), | support logic; impact depends on neighboring block and data flow. |
| 82 |     SetHitchTarget(f64), | support logic; impact depends on neighboring block and data flow. |
| 83 |     SetAuxValve(usize, f64), | support logic; impact depends on neighboring block and data flow. |
| 84 |     SetLoaderLift(f64), | support logic; impact depends on neighboring block and data flow. |
| 85 |     SetLoaderTilt(f64), | support logic; impact depends on neighboring block and data flow. |
| 86 |     // Hydraulics | comment/documentation; changing affects understanding and intent traceability. |
| 87 |     SetHitchJoystick(f64), | support logic; impact depends on neighboring block and data flow. |
| 88 |     // Autonomous | comment/documentation; changing affects understanding and intent traceability. |
| 89 |     EngageAD, | support logic; impact depends on neighboring block and data flow. |
| 90 |     DisengageAD, | support logic; impact depends on neighboring block and data flow. |
| 91 |     ToggleLka, | support logic; impact depends on neighboring block and data flow. |
| 92 |     AccSpeedSet(f64), | support logic; impact depends on neighboring block and data flow. |
| 93 |     #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 94 |     AccHeadwaySet(f64), | support logic; impact depends on neighboring block and data flow. |
| 95 |     AddWaypoint(f64, f64), | support logic; impact depends on neighboring block and data flow. |
| 96 |     ClearWaypoints, | support logic; impact depends on neighboring block and data flow. |
| 97 |     // Faults | comment/documentation; changing affects understanding and intent traceability. |
| 98 |     SelectFault(usize), | support logic; impact depends on neighboring block and data flow. |
| 99 |     InjectFault, | support logic; impact depends on neighboring block and data flow. |
| 100 |     ClearFaults, | support logic; impact depends on neighboring block and data flow. |
| 101 |     // Telematics | comment/documentation; changing affects understanding and intent traceability. |
| 102 |     StartOta, | support logic; impact depends on neighboring block and data flow. |
| 103 |     // Leak Lab | comment/documentation; changing affects understanding and intent traceability. |
| 104 |     LeakSelectCircuit(usize), | support logic; impact depends on neighboring block and data flow. |
| 105 |     LeakApplyManual, | support logic; impact depends on neighboring block and data flow. |
| 106 |     LeakPredictScenarios, | support logic; impact depends on neighboring block and data flow. |
| 107 |     LeakAddCustomCircuit, | support logic; impact depends on neighboring block and data flow. |
| 108 |     LeakExportReportCsv(String), | support logic; impact depends on neighboring block and data flow. |
| 109 |     LeakExportReportJson(String), | support logic; impact depends on neighboring block and data flow. |
| 110 |     LeakExportPredCsv(String), | support logic; impact depends on neighboring block and data flow. |
| 111 |     LeakExportPredJson(String), | support logic; impact depends on neighboring block and data flow. |
| 112 |     LeakExportCatalogMaterialsCsv(String), | support logic; impact depends on neighboring block and data flow.
| 113 |     LeakExportCatalogMaterialsJson(String), | support logic; impact depends on neighboring block and data flow.
```

```
| 114 |     LeakExportCatalogOilsCsv(String), | support logic; impact depends on neighboring block and data flow. |
| 115 |     LeakExportCatalogOilsJson(String), | support logic; impact depends on neighboring block and data flow. |
| 116 |     LeakCalibrateFromCsv(String), | support logic; impact depends on neighboring block and data flow. |
| 117 |     LeakExportCalibrationCsv(String), | support logic; impact depends on neighboring block and data flow. |
| 118 |     LeakExportCalibrationJson(String), | support logic; impact depends on neighboring block and data flow. |
| 119 |     LeakRunMonteCarlo, | support logic; impact depends on neighboring block and data flow. |
| 120 |     LeakClearScenarioOutputs, | support logic; impact depends on neighboring block and data flow. |
| 121 |     LeakResetSimulation, | support logic; impact depends on neighboring block and data flow. |
| 122 |     LeakApplyAndPredict, | support logic; impact depends on neighboring block and data flow. |
| 123 |     LeakApplyAndMonteCarlo, | support logic; impact depends on neighboring block and data flow. |
| 124 |     // CAN network controls | comment/documentation; changing affects understanding and intent traceability. |
| 125 |     CanInjectBitError(usize), | support logic; impact depends on neighboring block and data flow. |
| 126 |     CanInjectAckError(usize), | support logic; impact depends on neighboring block and data flow. |
| 127 |     CanInjectBusOff(usize), | support logic; impact depends on neighboring block and data flow. |
| 128 |     CanInjectBabbling(usize), | support logic; impact depends on neighboring block and data flow. |
| 129 |     CanClearBusInjections(usize), | support logic; impact depends on neighboring block and data flow. |
| 130 |     CanClearAllInjections, | support logic; impact depends on neighboring block and data flow. |
| 131 |     CanExportSnapshotCsv(String), | support logic; impact depends on neighboring block and data flow. |
| 132 |     CanExportSnapshotJson(String), | support logic; impact depends on neighboring block and data flow. |
| 133 |     // Simulation | comment/documentation; changing affects understanding and intent traceability. |
| 134 |     #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 135 |     Pause, | support logic; impact depends on neighboring block and data flow. |
| 136 |     #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 137 |     Resume, | support logic; impact depends on neighboring block and data flow. |
| 138 |     Reset, | support logic; impact depends on neighboring block and data flow. |
| 139 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 140 | (blank) | blank separator; no runtime behavior. |
| 141 | // ■■■
```

```
| 190 |             pressure_max_bar: 120.0, | support logic; impact depends on neighboring block and data flow. |
| 191 |             pressure_rupture_bar: 160.0, | support logic; impact depends on neighboring block and data flow. |
| 192 |             squeeze_pct: 18.0, | support logic; impact depends on neighboring block and data flow. |
| 193 |             compression_set_pct: 12.0, | support logic; impact depends on neighboring block and data flow. |
| 194 |             base_leak_area_mm2: 0.0005, | support logic; impact depends on neighboring block and data flow. |
| 195 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 196 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 197 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 198 | (blank) | blank separator; no runtime behavior. |
| 199 | #[derive(Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 200 | struct LeakCustomUi { | data layout contract; field changes can break callers/serde/tests. |
| 201 |     name: String, | support logic; impact depends on neighboring block and data flow. |
| 202 |     application: String, | support logic; impact depends on neighboring block and data flow. |
| 203 |     component_idx: usize, | support logic; impact depends on neighboring block and data flow. |
| 204 |     oil_type_idx: usize, | support logic; impact depends on neighboring block and data flow. |
| 205 |     seal_count: u32, | support logic; impact depends on neighboring block and data flow. |
| 206 |     piston_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 207 |     operation_pressure_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 208 |     min_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 209 |     mean_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 210 |     ideal_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 211 |     max_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 212 |     rupture_bar: f64, | support logic; impact depends on neighboring block and data flow. |
| 213 |     material_idx: usize, | support logic; impact depends on neighboring block and data flow. |
| 214 |     shore_a: f64, | support logic; impact depends on neighboring block and data flow. |
| 215 |     cross_section_mm: f64, | support logic; impact depends on neighboring block and data flow. |
| 216 |     squeeze_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 217 |     extrusion_gap_mm: f64, | support logic; impact depends on neighboring block and data flow. |
| 218 |     compression_set_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 219 |     design_life_hours: f64, | support logic; impact depends on neighboring block and data flow. |
| 220 |     base_leak_area_mm2: f64, | support logic; impact depends on neighboring block and data flow. |
| 221 |     discharge_coeff: f64, | support logic; impact depends on neighboring block and data flow. |
| 222 |     reservoir_volume_l: f64, | support logic; impact depends on neighboring block and data flow. |
| 223 |     support_lpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 224 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 225 | (blank) | blank separator; no runtime behavior. |
| 226 | impl Default for LeakCustomUi { | implementation binding; behavior semantics and trait conformance live here. |
| 227 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 228 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 229 |             name: "CUSTOM_01".into(), | support logic; impact depends on neighboring block and data flow. |
| 230 |             application: "Circuito custom de producao".into(), | support logic; impact depends on neighboring b
| 231 |             component_idx: 0, | support logic; impact depends on neighboring block and data flow. |
| 232 |             oil_type_idx: 0, | support logic; impact depends on neighboring block and data flow. |
| 233 |             seal_count: 4, | support logic; impact depends on neighboring block and data flow. |
| 234 |             piston_pressure_bar: 120.0, | support logic; impact depends on neighboring block and data flow. |
| 235 |             operation_pressure_bar: 95.0, | support logic; impact depends on neighboring block and data flow. |
| 236 |             min_bar: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 237 |             mean_bar: 70.0, | support logic; impact depends on neighboring block and data flow. |
| 238 |             ideal_bar: 95.0, | support logic; impact depends on neighboring block and data flow. |
| 239 |             max_bar: 130.0, | support logic; impact depends on neighboring block and data flow. |
| 240 |             rupture_bar: 180.0, | support logic; impact depends on neighboring block and data flow. |
| 241 |             material_idx: 0, | support logic; impact depends on neighboring block and data flow. |
| 242 |             shore_a: 80.0, | support logic; impact depends on neighboring block and data flow. |
| 243 |             cross_section_mm: 3.53, | support logic; impact depends on neighboring block and data flow. |
| 244 |             squeeze_pct: 18.0, | support logic; impact depends on neighboring block and data flow. |
| 245 |             extrusion_gap_mm: 0.12, | support logic; impact depends on neighboring block and data flow. |
| 246 |             compression_set_pct: 10.0, | support logic; impact depends on neighboring block and data flow. |
| 247 |             design_life_hours: 8000.0, | support logic; impact depends on neighboring block and data flow. |
| 248 |             base_leak_area_mm2: 0.0005, | support logic; impact depends on neighboring block and data flow. |
| 249 |             discharge_coeff: 0.58, | support logic; impact depends on neighboring block and data flow. |
| 250 |             reservoir_volume_l: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 251 |             support_lpm: 40.0, | support logic; impact depends on neighboring block and data flow. |
| 252 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 253 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 254 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 255 | (blank) | blank separator; no runtime behavior. |
| 256 | #[derive(PartialEq, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceabili
| 257 | enum CanMode { | state machine variants; changes ripple through matches and business logic. |
| 258 |     Signals, | support logic; impact depends on neighboring block and data flow. |
| 259 |     Trace, | support logic; impact depends on neighboring block and data flow. |
| 260 |     Network, | support logic; impact depends on neighboring block and data flow. |
| 261 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 262 | (blank) | blank separator; no runtime behavior. |
| 263 | #[derive(Clone, Copy, PartialEq, Debug)] | comment/documentation; changing affects understanding and intent tra
| 264 | pub enum EventLevel { | state machine variants; changes ripple through matches and business logic. |
| 265 |     Debug, | support logic; impact depends on neighboring block and data flow. |
```

```
| 266 |     Info, | support logic; impact depends on neighboring block and data flow. |
| 267 |     Ok, | support logic; impact depends on neighboring block and data flow. |
| 268 |     Warn, | support logic; impact depends on neighboring block and data flow. |
| 269 |     Critical, | support logic; impact depends on neighboring block and data flow. |
| 270 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 271 | (blank) | blank separator; no runtime behavior. |
| 272 | #[derive(Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 273 | struct AppEvent { | data layout contract; field changes can break callers/serde/tests. |
| 274 |     ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 275 |     source: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 276 |     msg: String, | support logic; impact depends on neighboring block and data flow. |
| 277 |     lvl: EventLevel, | support logic; impact depends on neighboring block and data flow. |
| 278 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 279 | (blank) | blank separator; no runtime behavior. |
| 280 | struct Signal { | data layout contract; field changes can break callers/serde/tests. |
| 281 |     pgn_name: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 282 |     sa_name: String, | support logic; impact depends on neighboring block and data flow. |
| 283 |     raw_id: u32, | support logic; impact depends on neighboring block and data flow. |
| 284 |     last_ts: f64, | support logic; impact depends on neighboring block and data flow. |
| 285 |     count: u64, | support logic; impact depends on neighboring block and data flow. |
| 286 |     period_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 287 |     decoded: Vec<(String, f64, String)>, | support logic; impact depends on neighboring block and data flow. |
| 288 |     fresh: bool, | support logic; impact depends on neighboring block and data flow. |
| 289 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 290 | (blank) | blank separator; no runtime behavior. |
| 291 | // ■■■
```

```
| 342 |     ev_min_level: EventLevel, | support logic; impact depends on neighboring block and data flow. |
| 343 | (blank) | blank separator; no runtime behavior. |
| 344 |     // Previous state for event detection | comment/documentation; changing affects understanding and intent tra
| 345 |     prev_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 346 |     prev_gear: String, | support logic; impact depends on neighboring block and data flow. |
| 347 |     prev_dtcs: usize, | support logic; impact depends on neighboring block and data flow. |
| 348 |     prev_abs: bool, | support logic; impact depends on neighboring block and data flow. |
| 349 |     prev_esp: EspCondition, | support logic; impact depends on neighboring block and data flow. |
| 350 |     prev_ign: IgnitionState, | support logic; impact depends on neighboring block and data flow. |
| 351 |     prev_regen: bool, | support logic; impact depends on neighboring block and data flow. |
| 352 |     prev_mode: String, | support logic; impact depends on neighboring block and data flow. |
| 353 | (blank) | blank separator; no runtime behavior. |
| 354 |     // Parameter editing — user-editable simulation params | comment/documentation; changing affects understandi
| 355 |     p_fuel: f64, | support logic; impact depends on neighboring block and data flow. |
| 356 |     p_def: f64, | support logic; impact depends on neighboring block and data flow. |
| 357 |     p_coolant: f64, | support logic; impact depends on neighboring block and data flow. |
| 358 |     p_oil_prs: f64, | support logic; impact depends on neighboring block and data flow. |
| 359 |     p_boost: f64, | support logic; impact depends on neighboring block and data flow. |
| 360 |     p_engine_h: f64, | support logic; impact depends on neighboring block and data flow. |
| 361 |     params_dirty: bool, | support logic; impact depends on neighboring block and data flow. |
| 362 | (blank) | blank separator; no runtime behavior. |
| 363 |     // Implements controls | comment/documentation; changing affects understanding and intent traceability. |
| 364 |     #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 365 |     pto_target_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 366 |     hitch_target: f64, | support logic; impact depends on neighboring block and data flow. |
| 367 |     aux_cmds: [f64; 4], | support logic; impact depends on neighboring block and data flow. |
| 368 | (blank) | blank separator; no runtime behavior. |
| 369 |     // AD path planning | comment/documentation; changing affects understanding and intent traceability. |
| 370 |     ad_waypoints: Vec<[f64; 2]>, // local ENU metres | support logic; impact depends on neighboring block and da
| 371 |     #[allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 372 |     ad_canvas_pos: f64, | support logic; impact depends on neighboring block and data flow. |
| 373 | (blank) | blank separator; no runtime behavior. |
| 374 |     // Fault panel | comment/documentation; changing affects understanding and intent traceability. |
| 375 |     fault_idx: usize, | support logic; impact depends on neighboring block and data flow. |
| 376 |     fault_feedback: String, | support logic; impact depends on neighboring block and data flow. |
| 377 |     fault_last_dtc_count: usize, | support logic; impact depends on neighboring block and data flow. |
| 378 | (blank) | blank separator; no runtime behavior. |
| 379 |     // Implements operator feedback | comment/documentation; changing affects understanding and intent traceabil
| 380 |     implement_feedback: String, | support logic; impact depends on neighboring block and data flow. |
| 381 | (blank) | blank separator; no runtime behavior. |
| 382 |     // V2X smoothed KPIs | comment/documentation; changing affects understanding and intent traceability. |
| 383 |     v2x_range_ema: f64, | support logic; impact depends on neighboring block and data flow. |
| 384 |     v2x_loss_ema: f64, | support logic; impact depends on neighboring block and data flow. |
| 385 |     v2x_lat_ema: f64, | support logic; impact depends on neighboring block and data flow. |
| 386 | (blank) | blank separator; no runtime behavior. |
| 387 |     // UDS console | comment/documentation; changing affects understanding and intent traceability. |
| 388 |     uds_input: String, | protocol or I/O critical; changes may impact external integration correctness. |
| 389 |     uds_sa: u8, | protocol or I/O critical; changes may impact external integration correctness. |
| 390 |     uds_flash_path: String, | protocol or I/O critical; changes may impact external integration correctness. |
| 391 |     uds_log: Vec<(bool, String)>, | protocol or I/O critical; changes may impact external integration correctnes
| 392 | (blank) | blank separator; no runtime behavior. |
| 393 |     // Leak physics lab | comment/documentation; changing affects understanding and intent traceability. |
| 394 |     leak_sel_idx: usize, | support logic; impact depends on neighboring block and data flow. |
| 395 |     leak_manual: LeakManualUi, | support logic; impact depends on neighboring block and data flow. |
| 396 |     leak_custom: LeakCustomUi, | support logic; impact depends on neighboring block and data flow. |
| 397 |     leak_horizon_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 398 |     leak_scenario_dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 399 |     leak_predictions: Vec<ScenarioPrediction>, | support logic; impact depends on neighboring block and data flo
| 400 |     leak_temporal_trace: VecDeque<(f64, f64, f64, f64)>, // (t, pressure, risk, leak) | support logic; impact de
| 401 |     leak_calibration_csv_path: String, | support logic; impact depends on neighboring block and data flow. |
| 402 |     leak_calibration_report: Option<CalibrationReport>, | support logic; impact depends on neighboring block and
| 403 |     leak_note: String, | support logic; impact depends on neighboring block and data flow. |
| 404 |     leak_view_yaw_deg: f32, | support logic; impact depends on neighboring block and data flow. |
| 405 |     leak_view_pitch_deg: f32, | support logic; impact depends on neighboring block and data flow. |
| 406 |     leak_view_zoom: f32, | support logic; impact depends on neighboring block and data flow. |
| 407 |     leak_timeback_window_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 408 |     leak_timeback_latest_first: bool, | support logic; impact depends on neighboring block and data flow. |
| 409 |     leak_timeback_stride: usize, | support logic; impact depends on neighboring block and data flow. |
| 410 | (blank) | blank separator; no runtime behavior. |
| 411 |     // Plots | comment/documentation; changing affects understanding and intent traceability. |
| 412 |     pl_rpm: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
| 413 |     pl_spd: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
| 414 |     pl_torque: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
| 415 |     pl_fuel: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
| 416 |     pl_coolant: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
| 417 |     pl_dpf: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
```

```
| 418 |     pl_boost: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
| 419 |     pl_hyd: Vec<[f64; 2]>, | support logic; impact depends on neighboring block and data flow. |
| 420 | (blank) | blank separator; no runtime behavior. |
| 421 |     ticks: u64, | support logic; impact depends on neighboring block and data flow. |
| 422 |     ux_show_guide: bool, | support logic; impact depends on neighboring block and data flow. |
| 423 |     ux_compact_mode: bool, | support logic; impact depends on neighboring block and data flow. |
| 424 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 425 | (blank) | blank separator; no runtime behavior. |
| 426 | impl App { | implementation binding; behavior semantics and trait conformance live here. |
| 427 |     fn new(cc: &eframe::CreationContext, hw_cfg: io::hw::HwConfig) -> Self { | function signature/body; changes
| 428 |         let mut vis = Visuals::dark(); | support logic; impact depends on neighboring block and data flow. |
| 429 |         vis.panel_fill = Color32::from_gray(14); | support logic; impact depends on neighboring block and data f
| 430 |         cc.egui_ctx.set_visuals(vis); | support logic; impact depends on neighboring block and data flow. |
| 431 | (blank) | blank separator; no runtime behavior. |
| 432 |         let mut bench = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flo
| 433 |         bench.tcm.set_direction(Direction::Forward); | support logic; impact depends on neighboring block and da
| 434 |         let fuel = bench.ecm.fuel_level_pct; | support logic; impact depends on neighboring block and data flow
| 435 |         let def = bench.ecm.def_level_pct; | support logic; impact depends on neighboring block and data flow.
| 436 |         let cool = bench.ecm.coolant_temp_c; | support logic; impact depends on neighboring block and data flow
| 437 |         let oil = bench.ecm.oil_pressure_kpa; | support logic; impact depends on neighboring block and data flow
| 438 |         let boost = bench.ecm.boost_pressure_kpa; | support logic; impact depends on neighboring block and data
| 439 |         let hrs = bench.ecm.engine_hours; | support logic; impact depends on neighboring block and data flow. |
| 440 | (blank) | blank separator; no runtime behavior. |
| 441 |         App { | support logic; impact depends on neighboring block and data flow. |
| 442 |             bench, | support logic; impact depends on neighboring block and data flow. |
| 443 |             hw_cfg, | support logic; impact depends on neighboring block and data flow. |
| 444 |             tab: Tab::Cluster, | support logic; impact depends on neighboring block and data flow. |
| 445 |             cmds: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 446 |             ecm_detected_sas: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 447 |             ecm_selected_idx: 0, | support logic; impact depends on neighboring block and data flow. |
| 448 |             ecm_connected: false, | support logic; impact depends on neighboring block and data flow. |
| 449 |             ecm_live_feed: None, | support logic; impact depends on neighboring block and data flow. |
| 450 |             ecm_live_status: "Idle. Press Detect to scan ECMs on the network.".into(), | support logic; impact
| 451 |             ecm_live_snapshot: io::ecm_params::EcmSnapshot::default(), | support logic; impact depends on neigh
| 452 |             ecm_live_last_update_ms: 0, | support logic; impact depends on neighboring block and data flow. |
| 453 |             ecm_live_frames_total: 0, | support logic; impact depends on neighboring block and data flow. |
| 454 |             throttle: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 455 |             brake: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 456 |             throttle_target: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 457 |             brake_target: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 458 |             throttle_cmd_last: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 459 |             brake_cmd_last: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 460 |             throttle_cmd_last_tick: 0, | support logic; impact depends on neighboring block and data flow. |
| 461 |             brake_cmd_last_tick: 0, | support logic; impact depends on neighboring block and data flow. |
| 462 |             can_mode: CanMode::Signals, | support logic; impact depends on neighboring block and data flow. |
| 463 |             can_freeze: false, | support logic; impact depends on neighboring block and data flow. |
| 464 |             can_filter: String::new(), | support logic; impact depends on neighboring block and data flow. |
| 465 |             sig_map: HashMap::new(), | support logic; impact depends on neighboring block and data flow. |
| 466 |             trace_snap: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 467 |             can_bus_idx: 0, | support logic; impact depends on neighboring block and data flow. |
| 468 |             can_note: String::new(), | support logic; impact depends on neighboring block and data flow. |
| 469 |             can_trace_tree: true, | support logic; impact depends on neighboring block and data flow. |
| 470 |             can_trace_limit: 500, | support logic; impact depends on neighboring block and data flow. |
| 471 |             can_signal_limit: 300, | support logic; impact depends on neighboring block and data flow. |
| 472 |             can_hide_stale: false, | support logic; impact depends on neighboring block and data flow. |
| 473 |             can_trace_update_every_ticks: 1, | support logic; impact depends on neighboring block and data flow
| 474 |             can_trace_tick_counter: 0, | support logic; impact depends on neighboring block and data flow. |
| 475 |             can_trace_pause_on_scroll: true, | support logic; impact depends on neighboring block and data flow
| 476 |             can_trace_pause_ticks_left: 0, | support logic; impact depends on neighboring block and data flow.
| 477 |             events: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 478 |             ev_pause: false, | support logic; impact depends on neighboring block and data flow. |
| 479 |             ev_filter: String::new(), | support logic; impact depends on neighboring block and data flow. |
| 480 |             ev_min_level: EventLevel::Info, | support logic; impact depends on neighboring block and data flow.
| 481 |             prev_rpm: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 482 |             prev_gear: "N".into(), | support logic; impact depends on neighboring block and data flow. |
| 483 |             prev_dtcs: 0, | support logic; impact depends on neighboring block and data flow. |
| 484 |             prev_abs: false, | support logic; impact depends on neighboring block and data flow. |
| 485 |             prev_esp: EspCondition::Neutral, | support logic; impact depends on neighboring block and data flow
| 486 |             prev_ign: IgnitionState::Off, | support logic; impact depends on neighboring block and data flow. |
| 487 |             prev_regen: false, | support logic; impact depends on neighboring block and data flow. |
| 488 |             prev_mode: String::new(), | support logic; impact depends on neighboring block and data flow. |
| 489 |             p_fuel: fuel, | support logic; impact depends on neighboring block and data flow. |
| 490 |             p_def: def, | support logic; impact depends on neighboring block and data flow. |
| 491 |             p_coolant: cool, | support logic; impact depends on neighboring block and data flow. |
| 492 |             p_oil_prs: oil, | support logic; impact depends on neighboring block and data flow. |
| 493 |             p_boost: boost, | support logic; impact depends on neighboring block and data flow. |
```

```
| 494 |             p_engine_h: hrs, | support logic; impact depends on neighboring block and data flow. |
| 495 |             params_dirty: false, | support logic; impact depends on neighboring block and data flow. |
| 496 |             pto_target_rpm: 540.0, | support logic; impact depends on neighboring block and data flow. |
| 497 |             hitch_target: 100.0, | support logic; impact depends on neighboring block and data flow. |
| 498 |             aux_cmds: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 499 |             ad_waypoints: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 500 |             ad_canvas_pos: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 501 |             fault_idx: 0, | support logic; impact depends on neighboring block and data flow. |
| 502 |             fault_feedback: String::new(), | support logic; impact depends on neighboring block and data flow.
| 503 |             fault_last_dtc_count: 0, | support logic; impact depends on neighboring block and data flow. |
| 504 |             implement_feedback: String::new(), | support logic; impact depends on neighboring block and data flo
| 505 |             v2x_range_ema: 300.0, | support logic; impact depends on neighboring block and data flow. |
| 506 |             v2x_loss_ema: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 507 |             v2x_lat_ema: 15.0, | support logic; impact depends on neighboring block and data flow. |
| 508 |             uds_input: "10 02".into(), | protocol or I/O critical; changes may impact external integration corre
| 509 |             uds_sa: 0x00, | protocol or I/O critical; changes may impact external integration correctness. |
| 510 |             uds_flash_path: String::new(), | protocol or I/O critical; changes may impact external integration c
| 511 |             uds_log: Vec::new(), | protocol or I/O critical; changes may impact external integration correctness
| 512 |             leak_sel_idx: 0, | support logic; impact depends on neighboring block and data flow. |
| 513 |             leak_manual: LeakManualUi::default(), | support logic; impact depends on neighboring block and data
| 514 |             leak_custom: LeakCustomUi::default(), | support logic; impact depends on neighboring block and data
| 515 |             leak_horizon_s: 600.0, | support logic; impact depends on neighboring block and data flow. |
| 516 |             leak_scenario_dt: 0.05, | support logic; impact depends on neighboring block and data flow. |
| 517 |             leak_predictions: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 518 |             leak_temporal_trace: VecDeque::new(), | support logic; impact depends on neighboring block and data
| 519 |             leak_calibration_csv_path: String::new(), | support logic; impact depends on neighboring block and c
| 520 |             leak_calibration_report: None, | support logic; impact depends on neighboring block and data flow.
| 521 |             leak_note: String::new(), | support logic; impact depends on neighboring block and data flow. |
| 522 |             leak_view_yaw_deg: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 523 |             leak_view_pitch_deg: 20.0, | support logic; impact depends on neighboring block and data flow. |
| 524 |             leak_view_zoom: 1.0, | support logic; impact depends on neighboring block and data flow. |
| 525 |             leak_timeback_window_s: 120.0, | support logic; impact depends on neighboring block and data flow.
| 526 |             leak_timeback_latest_first: false, | support logic; impact depends on neighboring block and data flo
| 527 |             leak_timeback_stride: 1, | support logic; impact depends on neighboring block and data flow. |
| 528 |             pl_rpm: vec![], | support logic; impact depends on neighboring block and data flow. |
| 529 |             pl_spd: vec![], | support logic; impact depends on neighboring block and data flow. |
| 530 |             pl_torque: vec![], | support logic; impact depends on neighboring block and data flow. |
| 531 |             pl_fuel: vec![], | support logic; impact depends on neighboring block and data flow. |
| 532 |             pl_coolant: vec![], | support logic; impact depends on neighboring block and data flow. |
| 533 |             pl_dpf: vec![], | support logic; impact depends on neighboring block and data flow. |
| 534 |             pl_boost: vec![], | support logic; impact depends on neighboring block and data flow. |
| 535 |             pl_hyd: vec![], | support logic; impact depends on neighboring block and data flow. |
| 536 |             ticks: 0, | support logic; impact depends on neighboring block and data flow. |
| 537 |             ux_show_guide: true, | support logic; impact depends on neighboring block and data flow. |
| 538 |             ux_compact_mode: false, | support logic; impact depends on neighboring block and data flow. |
| 539 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 540 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 541 | (blank) | blank separator; no runtime behavior. |
| 542 |     fn pick_save_path(default_name: &str, ext: &str) -> Option<String> { | function signature/body; changes affe
| 543 |         let default_dir = PathBuf::from("reports"); | support logic; impact depends on neighboring block and dat
| 544 |         let _ = std::fs::create_dir_all(&default_dir); | support logic; impact depends on neighboring block and
| 545 |         FileDialog::new() | support logic; impact depends on neighboring block and data flow. |
| 546 |             .set_directory(default_dir) | support logic; impact depends on neighboring block and data flow. |
| 547 |             .set_file_name(default_name) | support logic; impact depends on neighboring block and data flow. |
| 548 |             .add_filter(ext, &[ext]) | support logic; impact depends on neighboring block and data flow. |
| 549 |             .save_file() | support logic; impact depends on neighboring block and data flow. |
| 550 |             .map(\|p\| p.display().to_string()) | support logic; impact depends on neighboring block and data f
| 551 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 552 | (blank) | blank separator; no runtime behavior. |
| 553 |     fn pick_open_path(ext: &str) -> Option<String> { | function signature/body; changes affect API behavior and
| 554 |         FileDialog::new() | support logic; impact depends on neighboring block and data flow. |
| 555 |             .add_filter(ext, &[ext]) | support logic; impact depends on neighboring block and data flow. |
| 556 |             .pick_file() | support logic; impact depends on neighboring block and data flow. |
| 557 |             .map(\|p\| p.display().to_string()) | support logic; impact depends on neighboring block and data f
| 558 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 559 | (blank) | blank separator; no runtime behavior. |
| 560 |     fn refresh_mock_ecm_live_from_sim(&mut self) { | function signature/body; changes affect API behavior and c
| 561 |         let e = &self.bench.ecm; | support logic; impact depends on neighboring block and data flow. |
| 562 |         self.ecm_live_snapshot.engine_speed_rpm = Some(e.rpm); | support logic; impact depends on neighboring b
| 563 |         self.ecm_live_snapshot.accel_pedal_pct = Some(e.active_throttle); | support logic; impact depends on ne
| 564 |         self.ecm_live_snapshot.coolant_temp_c = Some(e.coolant_temp_c); | support logic; impact depends on neig
| 565 |         self.ecm_live_snapshot.fuel_temp_c = Some(e.fuel_temp_c); | support logic; impact depends on neighborin
| 566 |         self.ecm_live_snapshot.oil_pressure_kpa = Some(e.oil_pressure_kpa); | support logic; impact depends on
| 567 |         self.ecm_live_snapshot.last_seen_pgn = Some(61444); | support logic; impact depends on neighboring bloc
| 568 |         self.ecm_live_snapshot.source_address = Some(0x00); | support logic; impact depends on neighboring bloc
| 569 |         self.ecm_live_last_update_ms = now_ms(); | support logic; impact depends on neighboring block and data
```

```
| 570 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 571 | (blank) | blank separator; no runtime behavior. |
| 572 |     fn ecm_treeview(&self, ui: &mut Ui, mock_mode: bool) { | function signature/body; changes affect API behavio
| 573 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 574 |         ui.label(if mock_mode { | support logic; impact depends on neighboring block and data flow. |
| 575 |             "ECM Explorer (Mock/CAT ET style)" | support logic; impact depends on neighboring block and data flo
| 576 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 577 |             "ECM Explorer (CAT ET style)" | support logic; impact depends on neighboring block and data flow. |
| 578 |         }); | support logic; impact depends on neighboring block and data flow. |
| 579 | (blank) | blank separator; no runtime behavior. |
| 580 |         let nodes = ecm_mock_provider::build_mock_ecm_tree(&self.bench, &self.ecm_live_snapshot); | support logi
| 581 |         for node in nodes { | iteration logic; may alter timing, throughput, and safety constraints. |
| 582 |             CollapsingHeader::new(format!("{} (SA 0x{:02X})", node.name, node.source_address)) | support logic;
| 583 |                 .default_open(node.source_address == 0x00) | support logic; impact depends on neighboring block
| 584 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 585 |                     for f in node.functions { | iteration logic; may alter timing, throughput, and safety const
| 586 |                         CollapsingHeader::new(f.name).default_open(false).show(ui, \|ui\| { | support logic; imp
| 587 |                             Grid::new(format!("ecm_tree_{}_{}", node.source_address, f.name)) | support logic; i
| 588 |                                 .num_columns(3) | support logic; impact depends on neighboring block and data fl
| 589 |                                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data
| 590 |                                     for p in f.params { | iteration logic; may alter timing, throughput, and sa
| 591 |                                         ui.label(p.key); | support logic; impact depends on neighboring block a
| 592 |                                         ui.label(format!("{:.3}", p.value)); | support logic; impact depends on
| 593 |                                         ui.label(p.unit); | support logic; impact depends on neighboring block a
| 594 |                                         ui.end_row(); | support logic; impact depends on neighboring block and c
| 595 |                                     } | scope delimiter; structural only, but wrong placement changes ownership/
| 596 |                                 }); | support logic; impact depends on neighboring block and data flow. |
| 597 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 598 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 599 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 600 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 601 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 602 | (blank) | blank separator; no runtime behavior. |
| 603 |     fn oil_type_from_idx(idx: usize) -> OilType { | function signature/body; changes affect API behavior and co
| 604 |         OilType::all().get(idx).copied().unwrap_or(OilType::Custom) | support logic; impact depends on neighbori
| 605 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 606 | (blank) | blank separator; no runtime behavior. |
| 607 |     fn component_from_idx(idx: usize) -> CircuitComponent { | function signature/body; changes affect API behavi
| 608 |         match idx { | branch dispatcher; arm changes alter behavior class handling. |
| 609 |             0 => CircuitComponent::Oring, | support logic; impact depends on neighboring block and data flow. |
| 610 |             1 => CircuitComponent::Seal, | support logic; impact depends on neighboring block and data flow. |
| 611 |             _ => CircuitComponent::AcHose, | support logic; impact depends on neighboring block and data flow.
| 612 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 613 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 614 | (blank) | blank separator; no runtime behavior. |
| 615 |     fn material_from_idx(idx: usize) -> OringMaterial { | function signature/body; changes affect API behavior a
| 616 |         OringMaterial::all() | support logic; impact depends on neighboring block and data flow. |
| 617 |             .get(idx) | support logic; impact depends on neighboring block and data flow. |
| 618 |             .copied() | support logic; impact depends on neighboring block and data flow. |
| 619 |             .unwrap_or(OringMaterial::Nbr) | support logic; impact depends on neighboring block and data flow.
| 620 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 621 | (blank) | blank separator; no runtime behavior. |
| 622 |     fn idx_from_oil_type(o: OilType) -> usize { | function signature/body; changes affect API behavior and cont
| 623 |         OilType::all().iter().position(\|x\| *x == o).unwrap_or(0) | support logic; impact depends on neighborin
| 624 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 625 | (blank) | blank separator; no runtime behavior. |
| 626 |     fn sync_leak_manual_from_selected(&mut self) { | function signature/body; changes affect API behavior and co
| 627 |         if let Some(c) = self.bench.leak_rig.circuits.get(self.leak_sel_idx) { | runtime gate; condition change
| 628 |             self.leak_manual.oil_type_idx = Self::idx_from_oil_type(c.oil_type); | support logic; impact depend
| 629 |             self.leak_manual.piston_pressure_bar = c.piston_pressure_bar; | support logic; impact depends on ne
| 630 |             self.leak_manual.operation_pressure_bar = c.operation_pressure_bar; | support logic; impact depends
| 631 |             self.leak_manual.pressure_min_bar = c.pressure.min_bar; | support logic; impact depends on neighbori
| 632 |             self.leak_manual.pressure_mean_bar = c.pressure.mean_bar; | support logic; impact depends on neighbo
| 633 |             self.leak_manual.pressure_ideal_bar = c.pressure.ideal_bar; | support logic; impact depends on neigh
| 634 |             self.leak_manual.pressure_max_bar = c.pressure.max_bar; | support logic; impact depends on neighbori
| 635 |             self.leak_manual.pressure_rupture_bar = c.pressure.rupture_bar; | support logic; impact depends on n
| 636 |             self.leak_manual.squeeze_pct = c.spec.squeeze_pct; | support logic; impact depends on neighboring b
| 637 |             self.leak_manual.compression_set_pct = c.spec.compression_set_pct; | support logic; impact depends o
| 638 |             self.leak_manual.base_leak_area_mm2 = c.spec.base_leak_area_mm2; | support logic; impact depends on
| 639 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 640 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 641 | (blank) | blank separator; no runtime behavior. |
| 642 |     fn sanitize_leak_manual(&mut self) { | function signature/body; changes affect API behavior and control-flow
| 643 |         self.leak_manual.piston_pressure_bar = self.leak_manual.piston_pressure_bar.clamp(0.0, 1000.0); | suppo
| 644 |         self.leak_manual.operation_pressure_bar = self.leak_manual.operation_pressure_bar.clamp(0.0, 1000.0); |
| 645 |         self.leak_manual.pressure_min_bar = self.leak_manual.pressure_min_bar.max(0.0); | support logic; impact
```

```
| 646 |         self.leak_manual.pressure_mean_bar = self | support logic; impact depends on neighboring block and data
| 647 |             .leak_manual | support logic; impact depends on neighboring block and data flow. |
| 648 |             .pressure_mean_bar | support logic; impact depends on neighboring block and data flow. |
| 649 |             .max(self.leak_manual.pressure_min_bar); | support logic; impact depends on neighboring block and da
| 650 |         self.leak_manual.pressure_ideal_bar = self | support logic; impact depends on neighboring block and data
| 651 |             .leak_manual | support logic; impact depends on neighboring block and data flow. |
| 652 |             .pressure_ideal_bar | support logic; impact depends on neighboring block and data flow. |
| 653 |             .max(self.leak_manual.pressure_mean_bar); | support logic; impact depends on neighboring block and d
| 654 |         self.leak_manual.pressure_max_bar = self | support logic; impact depends on neighboring block and data f
| 655 |             .leak_manual | support logic; impact depends on neighboring block and data flow. |
| 656 |             .pressure_max_bar | support logic; impact depends on neighboring block and data flow. |
| 657 |             .max(self.leak_manual.pressure_ideal_bar); | support logic; impact depends on neighboring block and
| 658 |         self.leak_manual.pressure_rupture_bar = self | support logic; impact depends on neighboring block and da
| 659 |             .leak_manual | support logic; impact depends on neighboring block and data flow. |
| 660 |             .pressure_rupture_bar | support logic; impact depends on neighboring block and data flow. |
| 661 |             .max(self.leak_manual.pressure_max_bar + 0.1); | support logic; impact depends on neighboring block
| 662 |         self.leak_manual.squeeze_pct = self.leak_manual.squeeze_pct.clamp(5.0, 45.0); | support logic; impact de
| 663 |         self.leak_manual.compression_set_pct = self.leak_manual.compression_set_pct.clamp(0.0, 80.0); | support
| 664 |         self.leak_manual.base_leak_area_mm2 = self.leak_manual.base_leak_area_mm2.clamp(0.0, 0.1); | support log
| 665 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 666 | (blank) | blank separator; no runtime behavior. |
| 667 |     fn sanitize_leak_custom(&mut self) { | function signature/body; changes affect API behavior and control-flow
| 668 |         self.leak_custom.name = self.leak_custom.name.trim().to_string(); | support logic; impact depends on nei
| 669 |         self.leak_custom.application = self.leak_custom.application.trim().to_string(); | support logic; impact
| 670 |         self.leak_custom.seal_count = self.leak_custom.seal_count.clamp(1, 128); | support logic; impact depends
| 671 |         self.leak_custom.shore_a = self.leak_custom.shore_a.clamp(40.0, 95.0); | support logic; impact depends o
| 672 |         self.leak_custom.piston_pressure_bar = self.leak_custom.piston_pressure_bar.clamp(0.0, 1000.0); | suppo
| 673 |         self.leak_custom.operation_pressure_bar = self.leak_custom.operation_pressure_bar.clamp(0.0, 1000.0); |
| 674 |         self.leak_custom.min_bar = self.leak_custom.min_bar.max(0.0); | support logic; impact depends on neighbo
| 675 |         self.leak_custom.mean_bar = self.leak_custom.mean_bar.max(self.leak_custom.min_bar); | support logic; im
| 676 |         self.leak_custom.ideal_bar = self.leak_custom.ideal_bar.max(self.leak_custom.mean_bar); | support logic
| 677 |         self.leak_custom.max_bar = self.leak_custom.max_bar.max(self.leak_custom.ideal_bar); | support logic; im
| 678 |         self.leak_custom.rupture_bar = self.leak_custom.rupture_bar.max(self.leak_custom.max_bar + 0.1); | suppo
| 679 |         self.leak_custom.cross_section_mm = self.leak_custom.cross_section_mm.clamp(0.5, 20.0); | support logic
| 680 |         self.leak_custom.squeeze_pct = self.leak_custom.squeeze_pct.clamp(5.0, 45.0); | support logic; impact de
| 681 |         self.leak_custom.extrusion_gap_mm = self.leak_custom.extrusion_gap_mm.clamp(0.01, 3.0); | support logic
| 682 |         self.leak_custom.compression_set_pct = self.leak_custom.compression_set_pct.clamp(0.0, 80.0); | support
| 683 |         self.leak_custom.design_life_hours = self.leak_custom.design_life_hours.clamp(1.0, 200000.0); | support
| 684 |         self.leak_custom.base_leak_area_mm2 = self.leak_custom.base_leak_area_mm2.clamp(0.0, 0.1); | support log
| 685 |         self.leak_custom.discharge_coeff = self.leak_custom.discharge_coeff.clamp(0.05, 1.0); | support logic; i
| 686 |         self.leak_custom.reservoir_volume_l = self.leak_custom.reservoir_volume_l.clamp(0.1, 2000.0); | support
| 687 |         self.leak_custom.support_lpm = self.leak_custom.support_lpm.clamp(0.0, 5000.0); | support logic; impact
| 688 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 689 | (blank) | blank separator; no runtime behavior. |
| 690 |     fn validate_leak_manual(&self) -> Result<(), String> { | function signature/body; changes affect API behavio
| 691 |         if !(self.leak_manual.pressure_min_bar <= self.leak_manual.pressure_mean_bar | runtime gate; condition c
| 692 |             && self.leak_manual.pressure_mean_bar <= self.leak_manual.pressure_ideal_bar | support logic; impact
| 693 |             && self.leak_manual.pressure_ideal_bar <= self.leak_manual.pressure_max_bar | support logic; impact
| 694 |             && self.leak_manual.pressure_max_bar < self.leak_manual.pressure_rupture_bar) | support logic; impac
| 695 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 696 |             return Err("Envelope invalido: requer min <= mean <= ideal <= max < rupture".into()); | support logi
| 697 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 698 |         if self.leak_manual.base_leak_area_mm2 <= 0.0 { | runtime gate; condition changes can enable/disable cri
| 699 |             return Err("Base leak deve ser > 0".into()); | support logic; impact depends on neighboring block an
| 700 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 701 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 702 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 703 | (blank) | blank separator; no runtime behavior. |
| 704 |     fn validate_leak_custom(&self) -> Result<(), String> { | function signature/body; changes affect API behavio
| 705 |         if self.leak_custom.name.is_empty() { | runtime gate; condition changes can enable/disable critical path
| 706 |             return Err("Nome do circuito custom nao pode ser vazio".into()); | support logic; impact depends on
| 707 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 708 |         if !(self.leak_custom.min_bar <= self.leak_custom.mean_bar | runtime gate; condition changes can enable/
| 709 |             && self.leak_custom.mean_bar <= self.leak_custom.ideal_bar | support logic; impact depends on neighb
| 710 |             && self.leak_custom.ideal_bar <= self.leak_custom.max_bar | support logic; impact depends on neighbo
| 711 |             && self.leak_custom.max_bar < self.leak_custom.rupture_bar) | support logic; impact depends on neigh
| 712 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 713 |             return Err("Envelope custom invalido: requer min <= mean <= ideal <= max < rupture".into()); | suppo
| 714 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 715 |         if self.leak_custom.base_leak_area_mm2 <= 0.0 { | runtime gate; condition changes can enable/disable cri
| 716 |             return Err("Base leak custom deve ser > 0".into()); | support logic; impact depends on neighboring b
| 717 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 718 |         Ok(()) | support logic; impact depends on neighboring block and data flow. |
| 719 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 720 | (blank) | blank separator; no runtime behavior. |
| 721 |     fn build_manual_leak_params(&self) -> ManualCircuitParams { | function signature/body; changes affect API be
```

```
| 722 |         ManualCircuitParams { | support logic; impact depends on neighboring block and data flow. |
| 723 |             oil_type: Some(Self::oil_type_from_idx(self.leak_manual.oil_type_idx)), | support logic; impact depe
| 724 |             piston_pressure_bar: Some(self.leak_manual.piston_pressure_bar), | support logic; impact depends on
| 725 |             operation_pressure_bar: Some(self.leak_manual.operation_pressure_bar), | support logic; impact deper
| 726 |             pressure_min_bar: Some(self.leak_manual.pressure_min_bar), | support logic; impact depends on neighl
| 727 |             pressure_mean_bar: Some(self.leak_manual.pressure_mean_bar), | support logic; impact depends on neig
| 728 |             pressure_ideal_bar: Some(self.leak_manual.pressure_ideal_bar), | support logic; impact depends on ne
| 729 |             pressure_max_bar: Some(self.leak_manual.pressure_max_bar), | support logic; impact depends on neighl
| 730 |             pressure_rupture_bar: Some(self.leak_manual.pressure_rupture_bar), | support logic; impact depends o
| 731 |             oring_squeeze_pct: Some(self.leak_manual.squeeze_pct), | support logic; impact depends on neighborin
| 732 |             compression_set_pct: Some(self.leak_manual.compression_set_pct), | support logic; impact depends on
| 733 |             base_leak_area_mm2: Some(self.leak_manual.base_leak_area_mm2), | support logic; impact depends on ne
| 734 |             max_supported_temp_c: None, | support logic; impact depends on neighboring block and data flow. |
| 735 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 736 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 737 | (blank) | blank separator; no runtime behavior. |
| 738 |     fn apply_manual_to_selected_circuit(&mut self) -> Result<String, String> { | function signature/body; change
| 739 |         self.validate_leak_manual()?; | error propagation point; changes alter failure propagation semantics. |
| 740 |         let Some(c) = self.bench.leak_rig.circuits.get(self.leak_sel_idx) else { | support logic; impact depends
| 741 |             return Err("Nenhum circuito selecionado".into()); | support logic; impact depends on neighboring blo
| 742 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 743 |         let name = c.name.clone(); | support logic; impact depends on neighboring block and data flow. |
| 744 |         let ok = self | support logic; impact depends on neighboring block and data flow. |
| 745 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 746 |             .apply_leak_manual_params(&name, self.build_manual_leak_params()); | support logic; impact depends o
| 747 |         if ok { | runtime gate; condition changes can enable/disable critical paths. |
| 748 |             Ok(name) | support logic; impact depends on neighboring block and data flow. |
| 749 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 750 |             Err("Falha no backend ao aplicar parametros".into()) | support logic; impact depends on neighboring
| 751 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 752 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 753 | (blank) | blank separator; no runtime behavior. |
| 754 |     fn can_bus_from_idx(idx: usize) -> auto_breaking::BusId { | function signature/body; changes affect API beha
| 755 |         match idx { | branch dispatcher; arm changes alter behavior class handling. |
| 756 |             0 => auto_breaking::BusId::PowertrainHs, | support logic; impact depends on neighboring block and da
| 757 |             1 => auto_breaking::BusId::ChassisHs, | support logic; impact depends on neighboring block and data
| 758 |             2 => auto_breaking::BusId::BodyMs, | support logic; impact depends on neighboring block and data flo
| 759 |             3 => auto_breaking::BusId::IsoBus, | support logic; impact depends on neighboring block and data flo
| 760 |             _ => auto_breaking::BusId::Diagnostic, | support logic; impact depends on neighboring block and data
| 761 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 762 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 763 | (blank) | blank separator; no runtime behavior. |
| 764 |     fn parse_u32_filter(input: &str) -> Option<u32> { | function signature/body; changes affect API behavior and
| 765 |         let s = input.trim().to_lowercase(); | support logic; impact depends on neighboring block and data flow
| 766 |         if let Some(hex) = s.strip_prefix("0x") { | runtime gate; condition changes can enable/disable critical
| 767 |             u32::from_str_radix(hex, 16).ok() | support logic; impact depends on neighboring block and data flow
| 768 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 769 |             s.parse::<u32>().ok() | support logic; impact depends on neighboring block and data flow. |
| 770 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 771 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 772 | (blank) | blank separator; no runtime behavior. |
| 773 |     fn can_filter_match_signal(filter: &str, pgn: u32, sa: u8, sig: &Signal) -> bool { | function signature/body
| 774 |         let filt = filter.trim().to_lowercase(); | support logic; impact depends on neighboring block and data
| 775 |         if filt.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 776 |             return true; | support logic; impact depends on neighboring block and data flow. |
| 777 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 778 |         if let Some(v) = filt.strip_prefix("sa:") { | runtime gate; condition changes can enable/disable critica
| 779 |             return Self::parse_u32_filter(v) == Some(sa as u32); | support logic; impact depends on neighboring
| 780 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 781 |         if let Some(v) = filt.strip_prefix("pgn:") { | runtime gate; condition changes can enable/disable critic
| 782 |             return Self::parse_u32_filter(v) == Some(pgn); | support logic; impact depends on neighboring block
| 783 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 784 |         if let Some(v) = filt.strip_prefix("id:") { | runtime gate; condition changes can enable/disable critica
| 785 |             return Self::parse_u32_filter(v) == Some(sig.raw_id); | support logic; impact depends on neighborin
| 786 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 787 |         sig.pgn_name.to_lowercase().contains(&filt) | support logic; impact depends on neighboring block and da
| 788 |             \|\| sig.sa_name.to_lowercase().contains(&filt) | support logic; impact depends on neighboring bloc
| 789 |             \|\| format!("{:02x}", sa).contains(&filt) | support logic; impact depends on neighboring block and
| 790 |             \|\| format!("0x{:02x}", sa).contains(&filt) | support logic; impact depends on neighboring block a
| 791 |             \|\| format!("{:05}", pgn).contains(&filt) | support logic; impact depends on neighboring block and
| 792 |             \|\| format!("{:08x}", sig.raw_id).contains(&filt) | support logic; impact depends on neighboring b
| 793 |             \|\| sig | support logic; impact depends on neighboring block and data flow. |
| 794 |                 .decoded | support logic; impact depends on neighboring block and data flow. |
| 795 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 796 |                 .any(\|(name, _, _)\| name.to_lowercase().contains(&filt)) | support logic; impact depends on ne
| 797 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 798 | (blank) | blank separator; no runtime behavior. |
| 799 |     fn can_filter_match_trace(filter: &str, row: &TraceRow) -> bool { | function signature/body; changes affect
| 800 |         let filt = filter.trim().to_lowercase(); | support logic; impact depends on neighboring block and data f
| 801 |         if filt.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 802 |             return true; | support logic; impact depends on neighboring block and data flow. |
| 803 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 804 |         let (_ts, raw_id, sa, _dlc, hex, pgn_sa, decoded) = row; | support logic; impact depends on neighboring
| 805 |         if let Some(v) = filt.strip_prefix("sa:") { | runtime gate; condition changes can enable/disable critica
| 806 |             return Self::parse_u32_filter(v) == Some(*sa as u32); | support logic; impact depends on neighboring
| 807 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 808 |         if let Some(v) = filt.strip_prefix("id:") { | runtime gate; condition changes can enable/disable critica
| 809 |             return Self::parse_u32_filter(v) == Some(*raw_id); | support logic; impact depends on neighboring b
| 810 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 811 |         pgn_sa.to_lowercase().contains(&filt) | support logic; impact depends on neighboring block and data flo
| 812 |             \|\| decoded.to_lowercase().contains(&filt) | support logic; impact depends on neighboring block an
| 813 |             \|\| hex.to_lowercase().contains(&filt) | support logic; impact depends on neighboring block and da
| 814 |             \|\| format!("{:02x}", sa).contains(&filt) | support logic; impact depends on neighboring block and
| 815 |             \|\| format!("0x{:02x}", sa).contains(&filt) | support logic; impact depends on neighboring block a
| 816 |             \|\| format!("{:08x}", raw_id).contains(&filt) | support logic; impact depends on neighboring block
| 817 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 818 | (blank) | blank separator; no runtime behavior. |
| 819 |     fn ema(prev: f64, value: f64, alpha: f64) -> f64 { | function signature/body; changes affect API behavior a
| 820 |         prev + alpha * (value - prev) | support logic; impact depends on neighboring block and data flow. |
| 821 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 822 | (blank) | blank separator; no runtime behavior. |
| 823 |     fn update_v2x_kpis(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 824 |         self.v2x_range_ema = Self::ema(self.v2x_range_ema, self.bench.v2x.range_m, 0.10); | support logic; impa
| 825 |         self.v2x_loss_ema = Self::ema(self.v2x_loss_ema, self.bench.v2x.packet_loss_pct, 0.12); | support logic
| 826 |         self.v2x_lat_ema = Self::ema(self.v2x_lat_ema, self.bench.v2x.latency_ms, 0.12); | support logic; impac
| 827 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 828 | (blank) | blank separator; no runtime behavior. |
| 829 |     fn approach_f32(current: f32, target: f32, max_step: f32) -> f32 { | function signature/body; changes affec
| 830 |         if (target - current).abs() <= max_step { | runtime gate; condition changes can enable/disable critical
| 831 |             target | support logic; impact depends on neighboring block and data flow. |
| 832 |         } else if target > current { | support logic; impact depends on neighboring block and data flow. |
| 833 |             current + max_step | support logic; impact depends on neighboring block and data flow. |
| 834 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 835 |             current - max_step | support logic; impact depends on neighboring block and data flow. |
| 836 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 837 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 838 | (blank) | blank separator; no runtime behavior. |
| 839 |     fn shape_pedal(x: f32, deadzone: f32) -> f64 { | function signature/body; changes affect API behavior and c
| 840 |         let n = ((x - deadzone) / (1.0 - deadzone)).clamp(0.0, 1.0); | support logic; impact depends on neighbo
| 841 |         let shaped = n * n * (3.0 - 2.0 * n); // smoothstep | support logic; impact depends on neighboring bloc
| 842 |         shaped as f64 | support logic; impact depends on neighboring block and data flow. |
| 843 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 844 | (blank) | blank separator; no runtime behavior. |
| 845 |     // ■ Execute all queued commands ■■■
```

```
| 874 |                 // ECM overrides | comment/documentation; changing affects understanding and intent traceabilit
| 875 |                 Cmd::SetFuelLevel(v) => { | support logic; impact depends on neighboring block and data flow. |
| 876 |                     self.bench.ecm.fuel_level_pct = v.clamp(0.0, 100.0); | support logic; impact depends on neig
| 877 |                     self.p_fuel = v; | support logic; impact depends on neighboring block and data flow. |
| 878 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 879 |                 Cmd::SetDefLevel(v) => { | support logic; impact depends on neighboring block and data flow. |
| 880 |                     self.bench.ecm.def_level_pct = v.clamp(0.0, 100.0); | support logic; impact depends on neigh
| 881 |                     self.p_def = v; | support logic; impact depends on neighboring block and data flow. |
| 882 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 883 |                 Cmd::SetCoolantTemp(v) => { | support logic; impact depends on neighboring block and data flow.
| 884 |                     self.bench.ecm.coolant_temp_c = v.clamp(-40.0, 130.0); | support logic; impact depends on ne
| 885 |                     self.p_coolant = v; | support logic; impact depends on neighboring block and data flow. |
| 886 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 887 |                 Cmd::SetOilPressure(v) => { | support logic; impact depends on neighboring block and data flow.
| 888 |                     self.bench.ecm.oil_pressure_kpa = v.clamp(0.0, 800.0); | support logic; impact depends on ne
| 889 |                     self.p_oil_prs = v; | support logic; impact depends on neighboring block and data flow. |
| 890 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 891 |                 Cmd::SetBoostPressure(v) => { | support logic; impact depends on neighboring block and data flo
| 892 |                     self.bench.ecm.boost_pressure_kpa = v.clamp(0.0, 350.0); | support logic; impact depends on
| 893 |                     self.p_boost = v; | support logic; impact depends on neighboring block and data flow. |
| 894 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 895 |                 Cmd::SetEngineHours(v) => { | support logic; impact depends on neighboring block and data flow.
| 896 |                     self.bench.ecm.engine_hours = v.max(0.0); | support logic; impact depends on neighboring bl
| 897 |                     self.p_engine_h = v; | support logic; impact depends on neighboring block and data flow. |
| 898 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 899 |                 Cmd::ClearDtcs => self.bench.clear_faults(), | support logic; impact depends on neighboring blo
| 900 |                 // BCM | comment/documentation; changing affects understanding and intent traceability. |
| 901 |                 Cmd::ToggleWorkLights => self.bench.bcm.toggle_work_lights(), | support logic; impact depends o
| 902 |                 Cmd::ToggleRoadLights => self.bench.bcm.toggle_road_lights(), | support logic; impact depends o
| 903 |                 Cmd::ToggleBeacon => self.bench.bcm.toggle_beacon(), | support logic; impact depends on neighbo
| 904 |                 Cmd::Honk => self.bench.bcm.honk(), | support logic; impact depends on neighboring block and da
| 905 |                 Cmd::CycleWiper => self.bench.bcm.cycle_wiper(), | support logic; impact depends on neighboring
| 906 |                 Cmd::ToggleHvac => self.bench.bcm.toggle_hvac(), | support logic; impact depends on neighboring
| 907 |                 Cmd::ResetBcmFuses => self.bench.bcm.reset_all_fuses(), | support logic; impact depends on neig
| 908 |                 Cmd::ElectricalSetBranchFault(id, fault) => { | support logic; impact depends on neighboring bl
| 909 |                     self.bench.electrical.set_branch_fault(&id, fault) | support logic; impact depends on neigh
| 910 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 911 |                 Cmd::ElectricalSetControlFault(point, fault) => { | support logic; impact depends on neighborin
| 912 |                     self.bench.electrical.set_control_fault(point, fault) | support logic; impact depends on ne
| 913 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 914 |                 Cmd::ElectricalResetFaults => self.bench.electrical.reset_faults(), | support logic; impact dep
| 915 |                 // Implements | comment/documentation; changing affects understanding and intent traceability.
| 916 |                 Cmd::SetPtoMode(m) => { | support logic; impact depends on neighboring block and data flow. |
| 917 |                     self.bench.implement.pto_rear_enabled = m != PtoMode::Off; | support logic; impact depends
| 918 |                     self.bench.implement.pto_mode = m; | support logic; impact depends on neighboring block and
| 919 |                     self.implement_feedback = format!("Rear PTO mode -> {}", m); | support logic; impact depend
| 920 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 921 |                 Cmd::SetHitchTarget(v) => { | support logic; impact depends on neighboring block and data flow.
| 922 |                     self.bench.implement.hitch_target_pct = v.clamp(0.0, 100.0); | support logic; impact depend
| 923 |                     self.hitch_target = v; | support logic; impact depends on neighboring block and data flow.
| 924 |                     self.implement_feedback = | support logic; impact depends on neighboring block and data flo
| 925 |                         format!("Hitch target set -> {:.1}%", self.hitch_target); | support logic; impact depen
| 926 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 927 |                 Cmd::SetAuxValve(i, v) => { | support logic; impact depends on neighboring block and data flow.
| 928 |                     if i < 4 { | runtime gate; condition changes can enable/disable critical paths. |
| 929 |                         self.bench.hcm.set_aux_cmd(i, v); | support logic; impact depends on neighboring block a
| 930 |                         self.bench.implement.aux_banks[i].direction = if v > 0.1 { | support logic; impact depe
| 931 |                             auto_breaking::implement::ValveDirection::Extend | support logic; impact depends on
| 932 |                         } else if v < -0.1 { | support logic; impact depends on neighboring block and data flow
| 933 |                             auto_breaking::implement::ValveDirection::Retract | support logic; impact depends o
| 934 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 935 |                             auto_breaking::implement::ValveDirection::Neutral | support logic; impact depends o
| 936 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 937 |                         self.bench.implement.aux_banks[i].engaged = v.abs() > 0.05; | support logic; impact dep
| 938 |                         self.aux_cmds[i] = v; | support logic; impact depends on neighboring block and data flo
| 939 |                         self.implement_feedback = | support logic; impact depends on neighboring block and data
| 940 |                             format!("Aux bank {} command -> {:+.1}", i, self.aux_cmds[i]); | support logic; impa
| 941 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 942 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 943 |                 Cmd::SetLoaderLift(v) => { | support logic; impact depends on neighboring block and data flow.
| 944 |                     self.bench.loader_lift_cmd = v; | support logic; impact depends on neighboring block and da
| 945 |                     self.implement_feedback = format!("Loader lift cmd -> {:+.2}", v); | support logic; impact
| 946 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 947 |                 Cmd::SetLoaderTilt(v) => { | support logic; impact depends on neighboring block and data flow.
| 948 |                     self.bench.loader_tilt_cmd = v; | support logic; impact depends on neighboring block and da
| 949 |                     self.implement_feedback = format!("Loader tilt cmd -> {:+.2}", v); | support logic; impact
```

```
| 950 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 951 |                 Cmd::SetHitchJoystick(v) => { | support logic; impact depends on neighboring block and data flow
| 952 |                     self.bench.hitch_joystick = v; | support logic; impact depends on neighboring block and data
| 953 |                     self.implement_feedback = format!("Hitch joystick -> {:+.1}", v); | support logic; impact de
| 954 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 955 |                 // AD | comment/documentation; changing affects understanding and intent traceability. |
| 956 |                 Cmd::EngageAD => { | support logic; impact depends on neighboring block and data flow. |
| 957 |                     self.bench.ad.engage(self.bench.tcm.ground_speed_kmh); | support logic; impact depends on ne
| 958 |                     if self.bench.tcm.direction == Direction::Neutral | runtime gate; condition changes can enal
| 959 |                         \|\| self.bench.tcm.direction == Direction::Park | support logic; impact depends on neig
| 960 |                     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 961 |                         self.bench.tcm.set_direction(Direction::Forward); | support logic; impact depends on nei
| 962 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 963 |                     if self.bench.tcm.auto_mode != AutoShiftMode::Auto { | runtime gate; condition changes can e
| 964 |                         self.bench.tcm.toggle_auto(); | support logic; impact depends on neighboring block and e
| 965 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 966 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 967 |                 Cmd::DisengageAD => self.bench.ad.disengage(), | support logic; impact depends on neighboring bl
| 968 |                 Cmd::ToggleLka => self.bench.ad.toggle_lka(), | support logic; impact depends on neighboring blo
| 969 |                 Cmd::AccSpeedSet(v) => self.bench.ad.set_acc_speed(v), | support logic; impact depends on neighb
| 970 |                 Cmd::AccHeadwaySet(v) => self.bench.ad.set_headway(v), | support logic; impact depends on neighb
| 971 |                 Cmd::AddWaypoint(x, y) => self.ad_waypoints.push([x, y]), | support logic; impact depends on nei
| 972 |                 Cmd::ClearWaypoints => self.ad_waypoints.clear(), | support logic; impact depends on neighboring
| 973 |                 // Faults | comment/documentation; changing affects understanding and intent traceability. |
| 974 |                 Cmd::SelectFault(i) => { | support logic; impact depends on neighboring block and data flow. |
| 975 |                     self.fault_idx = i.min(FAULT_TYPES.len().saturating_sub(1)); | support logic; impact depends
| 976 |                     self.bench.selected_fault = FAULT_TYPES[self.fault_idx]; | support logic; impact depends on
| 977 |                     self.fault_feedback = | support logic; impact depends on neighboring block and data flow. |
| 978 |                         format!("Selected fault: {}", self.bench.selected_fault); | support logic; impact depend
| 979 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 980 |                 Cmd::InjectFault => { | support logic; impact depends on neighboring block and data flow. |
| 981 |                     self.bench.inject_fault(); | support logic; impact depends on neighboring block and data flo
| 982 |                     self.fault_last_dtc_count = self.bench.ecm.active_dtcs.len(); | support logic; impact depend
| 983 |                     self.fault_feedback = format!( | support logic; impact depends on neighboring block and data
| 984 |                         "Injected {}. Waiting ECM/DM1 propagation...", | support logic; impact depends on neighb
| 985 |                         self.bench.selected_fault | support logic; impact depends on neighboring block and data
| 986 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 987 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 988 |                 Cmd::ClearFaults => { | support logic; impact depends on neighboring block and data flow. |
| 989 |                     self.bench.clear_faults(); | support logic; impact depends on neighboring block and data flo
| 990 |                     self.fault_last_dtc_count = self.bench.ecm.active_dtcs.len(); | support logic; impact depend
| 991 |                     self.fault_feedback = "All injected faults cleared".into(); | support logic; impact depends
| 992 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 993 |                 // Telematics | comment/documentation; changing affects understanding and intent traceability.
| 994 |                 Cmd::StartOta => self.bench.telematics.start_ota(), | support logic; impact depends on neighbor
| 995 |                 // Leak Lab | comment/documentation; changing affects understanding and intent traceability. |
| 996 |                 Cmd::LeakSelectCircuit(i) => { | support logic; impact depends on neighboring block and data flo
| 997 |                     self.leak_sel_idx = i.min(self.bench.leak_rig.circuits.len().saturating_sub(1)); | support
| 998 |                     self.sync_leak_manual_from_selected(); | support logic; impact depends on neighboring block
| 999 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1000 |                 Cmd::LeakApplyManual => { | support logic; impact depends on neighboring block and data flow.
| 1001 |                     self.leak_note = match self.apply_manual_to_selected_circuit() { | support logic; impact de
| 1002 |                         Ok(name) => format!("Parametros aplicados com sucesso em {}", name), | support logic; i
| 1003 |                         Err(e) => format!("Falha ao aplicar parametros: {}", e), | support logic; impact depend
| 1004 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1005 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1006 |                 Cmd::LeakPredictScenarios => { | support logic; impact depends on neighboring block and data f
| 1007 |                     self.leak_predictions = self.bench.predict_leak_scenarios( | support logic; impact depends
| 1008 |                         self.leak_horizon_s, | support logic; impact depends on neighboring block and data flo
| 1009 |                         self.leak_scenario_dt.max(0.01), | support logic; impact depends on neighboring block a
| 1010 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 1011 |                     self.leak_note = format!("{} cenarios avaliados", self.leak_predictions.len()); | support
| 1012 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1013 |                 Cmd::LeakAddCustomCircuit => { | support logic; impact depends on neighboring block and data f
| 1014 |                     if let Err(e) = self.validate_leak_custom() { | runtime gate; condition changes can enable/
| 1015 |                         self.leak_note = format!("Falha ao adicionar circuito custom: {}", e); | support logic
| 1016 |                         continue; | support logic; impact depends on neighboring block and data flow. |
| 1017 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1018 |                     let c = LeakCircuit::new( | support logic; impact depends on neighboring block and data flo
| 1019 |                         &self.leak_custom.name, | support logic; impact depends on neighboring block and data
| 1020 |                         &self.leak_custom.application, | support logic; impact depends on neighboring block and
| 1021 |                         Self::component_from_idx(self.leak_custom.component_idx), | support logic; impact depe
| 1022 |                         Self::oil_type_from_idx(self.leak_custom.oil_type_idx), | support logic; impact depends
| 1023 |                         self.leak_custom.seal_count.max(1), | support logic; impact depends on neighboring bloc
| 1024 |                         self.leak_custom.piston_pressure_bar, | support logic; impact depends on neighboring b
| 1025 |                         self.leak_custom.operation_pressure_bar, | support logic; impact depends on neighboring
```

```
| 1026 |                         PressureEnvelope { | support logic; impact depends on neighboring block and data flow.
| 1027 |                             min_bar: self.leak_custom.min_bar, | support logic; impact depends on neighboring b
| 1028 |                             mean_bar: self.leak_custom.mean_bar, | support logic; impact depends on neighboring
| 1029 |                             ideal_bar: self.leak_custom.ideal_bar, | support logic; impact depends on neighbori
| 1030 |                             max_bar: self.leak_custom.max_bar, | support logic; impact depends on neighboring b
| 1031 |                             rupture_bar: self | support logic; impact depends on neighboring block and data flo
| 1032 |                                 .leak_custom | support logic; impact depends on neighboring block and data flow
| 1033 |                                 .rupture_bar | support logic; impact depends on neighboring block and data flow
| 1034 |                                 .max(self.leak_custom.max_bar + 0.1), | support logic; impact depends on neighb
| 1035 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 1036 |                         OringSpec { | support logic; impact depends on neighboring block and data flow. |
| 1037 |                             id_tag: format!("{}-custom", self.leak_custom.name), | support logic; impact depend
| 1038 |                             material: Self::material_from_idx(self.leak_custom.material_idx), | support logic;
| 1039 |                             shore_a: self.leak_custom.shore_a, | support logic; impact depends on neighboring b
| 1040 |                             cross_section_mm: self.leak_custom.cross_section_mm, | support logic; impact depend
| 1041 |                             squeeze_pct: self.leak_custom.squeeze_pct, | support logic; impact depends on neigh
| 1042 |                             extrusion_gap_mm: self.leak_custom.extrusion_gap_mm, | support logic; impact depend
| 1043 |                             compression_set_pct: self.leak_custom.compression_set_pct, | support logic; impact
| 1044 |                             design_life_hours: self.leak_custom.design_life_hours, | support logic; impact depe
| 1045 |                             base_leak_area_mm2: self.leak_custom.base_leak_area_mm2, | support logic; impact de
| 1046 |                             discharge_coeff: self.leak_custom.discharge_coeff, | support logic; impact depends
| 1047 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 1048 |                         self.leak_custom.reservoir_volume_l, | support logic; impact depends on neighboring blo
| 1049 |                         self.leak_custom.support_lpm, | support logic; impact depends on neighboring block and
| 1050 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 1051 |                     self.bench.add_custom_leak_circuit(c); | support logic; impact depends on neighboring bloc
| 1052 |                     self.leak_sel_idx = self.bench.leak_rig.circuits.len().saturating_sub(1); | support logic;
| 1053 |                     self.sync_leak_manual_from_selected(); | support logic; impact depends on neighboring bloc
| 1054 |                     self.leak_note = "Circuito custom adicionado".into(); | support logic; impact depends on ne
| 1055 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1056 |                 Cmd::LeakExportReportCsv(path) => { | support logic; impact depends on neighboring block and da
| 1057 |                     self.leak_note = match self.bench.export_leak_report_csv(&path) { | support logic; impact d
| 1058 |                         Ok(_) => format!("CSV salvo em {}", path), | support logic; impact depends on neighbori
| 1059 |                         Err(e) => format!("Falha export CSV: {}", e), | support logic; impact depends on neighb
| 1060 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1061 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1062 |                 Cmd::LeakExportReportJson(path) => { | support logic; impact depends on neighboring block and d
| 1063 |                     self.leak_note = match self.bench.export_leak_report_json(&path) { | support logic; impact
| 1064 |                         Ok(_) => format!("JSON salvo em {}", path), | support logic; impact depends on neighbo
| 1065 |                         Err(e) => format!("Falha export JSON: {}", e), | support logic; impact depends on neigh
| 1066 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1067 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1068 |                 Cmd::LeakExportPredCsv(path) => { | support logic; impact depends on neighboring block and data
| 1069 |                     self.leak_note = match self | support logic; impact depends on neighboring block and data f
| 1070 |                         .bench | support logic; impact depends on neighboring block and data flow. |
| 1071 |                         .export_leak_predictions_csv(&path, &self.leak_predictions) | support logic; impact dep
| 1072 |                     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1073 |                         Ok(_) => format!("Pred CSV salvo em {}", path), | support logic; impact depends on neig
| 1074 |                         Err(e) => format!("Falha export pred CSV: {}", e), | support logic; impact depends on n
| 1075 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1076 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1077 |                 Cmd::LeakExportPredJson(path) => { | support logic; impact depends on neighboring block and dat
| 1078 |                     self.leak_note = match self | support logic; impact depends on neighboring block and data f
| 1079 |                         .bench | support logic; impact depends on neighboring block and data flow. |
| 1080 |                         .export_leak_predictions_json(&path, &self.leak_predictions) | support logic; impact de
| 1081 |                     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1082 |                         Ok(_) => format!("Pred JSON salvo em {}", path), | support logic; impact depends on nei
| 1083 |                         Err(e) => format!("Falha export pred JSON: {}", e), | support logic; impact depends on
| 1084 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1085 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1086 |                 Cmd::LeakExportCatalogMaterialsCsv(path) => { | support logic; impact depends on neighboring bl
| 1087 |                     self.leak_note = match self.bench.leak_rig.export_material_catalog_csv(&path) { | support
| 1088 |                         Ok(_) => format!("Catalogo materiais CSV salvo em {}", path), | support logic; impact d
| 1089 |                         Err(e) => format!("Falha export catalogo materiais CSV: {}", e), | support logic; impac
| 1090 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1091 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1092 |                 Cmd::LeakExportCatalogMaterialsJson(path) => { | support logic; impact depends on neighboring b
| 1093 |                     self.leak_note = | support logic; impact depends on neighboring block and data flow. |
| 1094 |                         match self.bench.leak_rig.export_material_catalog_json(&path) { | branch dispatcher; a
| 1095 |                             Ok(_) => format!("Catalogo materiais JSON salvo em {}", path), | support logic; imp
| 1096 |                             Err(e) => format!("Falha export catalogo materiais JSON: {}", e), | support logic;
| 1097 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 1098 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1099 |                 Cmd::LeakExportCatalogOilsCsv(path) => { | support logic; impact depends on neighboring block a
| 1100 |                     self.leak_note = match self.bench.leak_rig.export_oil_catalog_csv(&path) { | support logic
| 1101 |                         Ok(_) => format!("Catalogo oleos CSV salvo em {}", path), | support logic; impact depe
```

```
| 1102 |                         Err(e) => format!("Falha export catalogo oleos CSV: {}", e), | support logic; impact de
| 1103 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1104 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1105 |                 Cmd::LeakExportCatalogOilsJson(path) => { | support logic; impact depends on neighboring block
| 1106 |                     self.leak_note = match self.bench.leak_rig.export_oil_catalog_json(&path) { | support logic
| 1107 |                         Ok(_) => format!("Catalogo oleos JSON salvo em {}", path), | support logic; impact depe
| 1108 |                         Err(e) => format!("Falha export catalogo oleos JSON: {}", e), | support logic; impact d
| 1109 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1110 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1111 |                 Cmd::LeakCalibrateFromCsv(path) => { | support logic; impact depends on neighboring block and d
| 1112 |                     self.leak_note = match self.bench.calibrate_leak_model_from_csv(&path) { | support logic; i
| 1113 |                         Ok(report) => { | support logic; impact depends on neighboring block and data flow. |
| 1114 |                             let circuits = report.calibrated_circuits; | support logic; impact depends on neigh
| 1115 |                             let samples = report.total_samples; | support logic; impact depends on neighboring
| 1116 |                             self.leak_calibration_report = Some(report); | support logic; impact depends on nei
| 1117 |                             self.leak_calibration_csv_path = path.clone(); | support logic; impact depends on n
| 1118 |                             format!( | support logic; impact depends on neighboring block and data flow. |
| 1119 |                                 "Calibracao concluida: {} circuitos, {} amostras", | support logic; impact depe
| 1120 |                                 circuits, samples | support logic; impact depends on neighboring block and data
| 1121 |                             ) | support logic; impact depends on neighboring block and data flow. |
| 1122 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1123 |                         Err(e) => format!("Falha calibracao CSV: {}", e), | support logic; impact depends on ne
| 1124 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1125 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1126 |                 Cmd::LeakExportCalibrationCsv(path) => { | support logic; impact depends on neighboring block a
| 1127 |                     self.leak_note = match self.leak_calibration_report.as_ref() { | support logic; impact depe
| 1128 |                         Some(report) => match self | support logic; impact depends on neighboring block and dat
| 1129 |                             .bench | support logic; impact depends on neighboring block and data flow. |
| 1130 |                             .export_leak_calibration_report_csv(&path, report) | support logic; impact depends
| 1131 |                         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1132 |                             Ok(_) => format!("Calibracao CSV salva em {}", path), | support logic; impact deper
| 1133 |                             Err(e) => format!("Falha export calibracao CSV: {}", e), | support logic; impact de
| 1134 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 1135 |                         None => "Execute calibracao antes de exportar".into(), | support logic; impact depends
| 1136 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1137 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1138 |                 Cmd::LeakExportCalibrationJson(path) => { | support logic; impact depends on neighboring block
| 1139 |                     self.leak_note = match self.leak_calibration_report.as_ref() { | support logic; impact depe
| 1140 |                         Some(report) => match self | support logic; impact depends on neighboring block and dat
| 1141 |                             .bench | support logic; impact depends on neighboring block and data flow. |
| 1142 |                             .export_leak_calibration_report_json(&path, report) | support logic; impact depends
| 1143 |                         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1144 |                             Ok(_) => format!("Calibracao JSON salva em {}", path), | support logic; impact depe
| 1145 |                             Err(e) => format!("Falha export calibracao JSON: {}", e), | support logic; impact d
| 1146 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 1147 |                         None => "Execute calibracao antes de exportar".into(), | support logic; impact depends
| 1148 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1149 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1150 |                 Cmd::LeakRunMonteCarlo => { | support logic; impact depends on neighboring block and data flow
| 1151 |                     self.leak_predictions = self.bench.monte_carlo_leak_predictions( | support logic; impact de
| 1152 |                         120, | support logic; impact depends on neighboring block and data flow. |
| 1153 |                         self.leak_horizon_s, | support logic; impact depends on neighboring block and data flow
| 1154 |                         self.leak_scenario_dt.max(0.01), | support logic; impact depends on neighboring block a
| 1155 |                         25.0, | support logic; impact depends on neighboring block and data flow. |
| 1156 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 1157 |                     self.leak_note = format!( | support logic; impact depends on neighboring block and data flo
| 1158 |                         "Monte Carlo concluido: {} amostras", | support logic; impact depends on neighboring b
| 1159 |                         self.leak_predictions.len() | support logic; impact depends on neighboring block and da
| 1160 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 1161 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1162 |                 Cmd::LeakClearScenarioOutputs => { | support logic; impact depends on neighboring block and dat
| 1163 |                     self.leak_predictions.clear(); | support logic; impact depends on neighboring block and dat
| 1164 |                     self.leak_temporal_trace.clear(); | support logic; impact depends on neighboring block and
| 1165 |                     self.leak_note = "Saida de simulacao limpa. Pronto para nova rodada.".into(); | support log
| 1166 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1167 |                 Cmd::LeakResetSimulation => { | support logic; impact depends on neighboring block and data flo
| 1168 |                     self.bench.leak_rig.reset(); | support logic; impact depends on neighboring block and data
| 1169 |                     self.bench.leak_reports.clear(); | support logic; impact depends on neighboring block and d
| 1170 |                     self.leak_predictions.clear(); | support logic; impact depends on neighboring block and dat
| 1171 |                     self.leak_temporal_trace.clear(); | support logic; impact depends on neighboring block and
| 1172 |                     self.leak_calibration_report = None; | support logic; impact depends on neighboring block a
| 1173 |                     self.leak_note = | support logic; impact depends on neighboring block and data flow. |
| 1174 |                         "Leak Lab resetado: estado fisico, alertas, historico e cenarios limpos." | support log
| 1175 |                             .into(); | support logic; impact depends on neighboring block and data flow. |
| 1176 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1177 |                 Cmd::LeakApplyAndPredict => { | support logic; impact depends on neighboring block and data flo
```

```
| 1178 |                     match self.apply_manual_to_selected_circuit() { | branch dispatcher; arm changes alter beha
| 1179 |                         Ok(_) => { | support logic; impact depends on neighboring block and data flow. |
| 1180 |                             self.leak_temporal_trace.clear(); | support logic; impact depends on neighboring bl
| 1181 |                             self.leak_predictions = self.bench.predict_leak_scenarios( | support logic; impact
| 1182 |                                 self.leak_horizon_s, | support logic; impact depends on neighboring block and d
| 1183 |                                 self.leak_scenario_dt.max(0.01), | support logic; impact depends on neighboring
| 1184 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 1185 |                             self.leak_note = | support logic; impact depends on neighboring block and data flow
| 1186 |                                 format!("Nova rodada: parametros aplicados + {} cenarios", self.leak_prediction
| 1187 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1188 |                         Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 1189 |                             self.leak_note = format!("Nova rodada: falha ao aplicar parametros: {}", e); | supp
| 1190 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1191 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1192 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1193 |                 Cmd::LeakApplyAndMonteCarlo => { | support logic; impact depends on neighboring block and data
| 1194 |                     match self.apply_manual_to_selected_circuit() { | branch dispatcher; arm changes alter beha
| 1195 |                         Ok(_) => { | support logic; impact depends on neighboring block and data flow. |
| 1196 |                             self.leak_temporal_trace.clear(); | support logic; impact depends on neighboring bl
| 1197 |                             self.leak_predictions = self.bench.monte_carlo_leak_predictions( | support logic; i
| 1198 |                                 120, | support logic; impact depends on neighboring block and data flow. |
| 1199 |                                 self.leak_horizon_s, | support logic; impact depends on neighboring block and d
| 1200 |                                 self.leak_scenario_dt.max(0.01), | support logic; impact depends on neighboring
| 1201 |                                 25.0, | support logic; impact depends on neighboring block and data flow. |
| 1202 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 1203 |                             self.leak_note = | support logic; impact depends on neighboring block and data flow
| 1204 |                                 format!("Nova rodada MC: parametros aplicados + {} amostras", self.leak_predic
| 1205 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1206 |                         Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 1207 |                             self.leak_note = format!("Nova rodada MC: falha ao aplicar parametros: {}", e); | s
| 1208 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1209 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1210 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1211 |                 Cmd::CanInjectBitError(idx) => { | support logic; impact depends on neighboring block and data
| 1212 |                     let bus = Self::can_bus_from_idx(idx); | support logic; impact depends on neighboring block
| 1213 |                     self.bench | support logic; impact depends on neighboring block and data flow. |
| 1214 |                         .can_net | support logic; impact depends on neighboring block and data flow. |
| 1215 |                         .inject_error_once(bus, auto_breaking::CanErrorKind::Bit); | support logic; impact depe
| 1216 |                     self.can_note = format!("Bit error armado em {}", bus); | support logic; impact depends on
| 1217 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1218 |                 Cmd::CanInjectAckError(idx) => { | support logic; impact depends on neighboring block and data
| 1219 |                     let bus = Self::can_bus_from_idx(idx); | support logic; impact depends on neighboring block
| 1220 |                     self.bench | support logic; impact depends on neighboring block and data flow. |
| 1221 |                         .can_net | support logic; impact depends on neighboring block and data flow. |
| 1222 |                         .inject_error_once(bus, auto_breaking::CanErrorKind::Ack); | support logic; impact depe
| 1223 |                     self.can_note = format!("ACK error armado em {}", bus); | support logic; impact depends on
| 1224 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1225 |                 Cmd::CanInjectBusOff(idx) => { | support logic; impact depends on neighboring block and data f
| 1226 |                     let bus = Self::can_bus_from_idx(idx); | support logic; impact depends on neighboring block
| 1227 |                     self.bench.can_net.inject_bus_off_once(bus); | support logic; impact depends on neighboring
| 1228 |                     self.can_note = format!("Bus-Off armado em {}", bus); | support logic; impact depends on ne
| 1229 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1230 |                 Cmd::CanInjectBabbling(idx) => { | support logic; impact depends on neighboring block and data
| 1231 |                     let bus = Self::can_bus_from_idx(idx); | support logic; impact depends on neighboring block
| 1232 |                     self.bench | support logic; impact depends on neighboring block and data flow. |
| 1233 |                         .can_net | support logic; impact depends on neighboring block and data flow. |
| 1234 |                         .inject_error_once(bus, auto_breaking::CanErrorKind::BablingIdiot); | support logic; im
| 1235 |                     self.can_note = format!("Babbling armado em {}", bus); | support logic; impact depends on
| 1236 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1237 |                 Cmd::CanClearBusInjections(idx) => { | support logic; impact depends on neighboring block and d
| 1238 |                     let bus = Self::can_bus_from_idx(idx); | support logic; impact depends on neighboring block
| 1239 |                     self.bench.can_net.clear_injections(Some(bus)); | support logic; impact depends on neighbo
| 1240 |                     self.can_note = format!("Injecoes limpas em {}", bus); | support logic; impact depends on
| 1241 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1242 |                 Cmd::CanClearAllInjections => { | support logic; impact depends on neighboring block and data
| 1243 |                     self.bench.can_net.clear_injections(None); | support logic; impact depends on neighboring
| 1244 |                     self.can_note = "Injecoes limpas em todos os barramentos".into(); | support logic; impact d
| 1245 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1246 |                 Cmd::CanExportSnapshotCsv(path) => { | support logic; impact depends on neighboring block and d
| 1247 |                     self.can_note = match self.bench.export_can_snapshot_csv(&path) { | support logic; impact d
| 1248 |                         Ok(_) => format!("Snapshot CSV salvo em {}", path), | support logic; impact depends on
| 1249 |                         Err(e) => format!("Falha snapshot CSV: {}", e), | support logic; impact depends on neig
| 1250 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1251 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1252 |                 Cmd::CanExportSnapshotJson(path) => { | support logic; impact depends on neighboring block and
| 1253 |                     self.can_note = match self.bench.export_can_snapshot_json(&path) { | support logic; impact
```

```
| 1254 |                         Ok(_) => format!("Snapshot JSON salvo em {}", path), | support logic; impact depends on
| 1255 |                         Err(e) => format!("Falha snapshot JSON: {}", e), | support logic; impact depends on nei
| 1256 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1257 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1258 |                 // Simulation | comment/documentation; changing affects understanding and intent traceability.
| 1259 |                 Cmd::Reset => { | support logic; impact depends on neighboring block and data flow. |
| 1260 |                     self.bench.reset(); | support logic; impact depends on neighboring block and data flow. |
| 1261 |                     self.bench.tcm.set_direction(Direction::Forward); | support logic; impact depends on neighb
| 1262 |                     self.throttle = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1263 |                     self.brake = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 1264 |                     self.throttle_target = 0.0; | support logic; impact depends on neighboring block and data
| 1265 |                     self.brake_target = 0.0; | support logic; impact depends on neighboring block and data flo
| 1266 |                     self.throttle_cmd_last = 0.0; | support logic; impact depends on neighboring block and data
| 1267 |                     self.brake_cmd_last = 0.0; | support logic; impact depends on neighboring block and data f
| 1268 |                     self.throttle_cmd_last_tick = self.ticks; | support logic; impact depends on neighboring b
| 1269 |                     self.brake_cmd_last_tick = self.ticks; | support logic; impact depends on neighboring bloc
| 1270 |                     self.sig_map.clear(); | support logic; impact depends on neighboring block and data flow.
| 1271 |                     self.events.clear(); | support logic; impact depends on neighboring block and data flow. |
| 1272 |                     self.ticks = 0; | support logic; impact depends on neighboring block and data flow. |
| 1273 |                     for v in [ | iteration logic; may alter timing, throughput, and safety constraints. |
| 1274 |                         &mut self.pl_rpm, | support logic; impact depends on neighboring block and data flow.
| 1275 |                         &mut self.pl_spd, | support logic; impact depends on neighboring block and data flow.
| 1276 |                         &mut self.pl_torque, | support logic; impact depends on neighboring block and data flo
| 1277 |                         &mut self.pl_fuel, | support logic; impact depends on neighboring block and data flow.
| 1278 |                         &mut self.pl_coolant, | support logic; impact depends on neighboring block and data fl
| 1279 |                         &mut self.pl_dpf, | support logic; impact depends on neighboring block and data flow.
| 1280 |                         &mut self.pl_boost, | support logic; impact depends on neighboring block and data flow
| 1281 |                         &mut self.pl_hyd, | support logic; impact depends on neighboring block and data flow.
| 1282 |                     ] { | support logic; impact depends on neighboring block and data flow. |
| 1283 |                         v.clear(); | support logic; impact depends on neighboring block and data flow. |
| 1284 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1285 |                     self.uds_log.clear(); | protocol or I/O critical; changes may impact external integration
| 1286 |                     self.ad_waypoints.clear(); | support logic; impact depends on neighboring block and data f
| 1287 |                     self.leak_predictions.clear(); | support logic; impact depends on neighboring block and da
| 1288 |                     self.leak_note.clear(); | support logic; impact depends on neighboring block and data flow
| 1289 |                     self.can_note.clear(); | support logic; impact depends on neighboring block and data flow.
| 1290 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1291 |                 Cmd::Pause \| Cmd::Resume => {} // UI-only, no bench mutation needed | support logic; impact de
| 1292 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1293 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1294 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1295 | (blank) | blank separator; no runtime behavior. |
| 1296 |     // ■ Sync parameter panel from simulation (when not dirty) ■■■■■■■■■■■■■■■■ | comment/documentation; char
| 1297 |     fn sync_params_from_bench(&mut self) { | function signature/body; changes affect API behavior and control-
| 1298 |         if !self.params_dirty { | runtime gate; condition changes can enable/disable critical paths. |
| 1299 |             self.p_fuel = self.bench.ecm.fuel_level_pct; | support logic; impact depends on neighboring block a
| 1300 |             self.p_def = self.bench.ecm.def_level_pct; | support logic; impact depends on neighboring block and
| 1301 |             self.p_coolant = self.bench.ecm.coolant_temp_c; | support logic; impact depends on neighboring bloc
| 1302 |             self.p_oil_prs = self.bench.ecm.oil_pressure_kpa; | support logic; impact depends on neighboring b
| 1303 |             self.p_boost = self.bench.ecm.boost_pressure_kpa; | support logic; impact depends on neighboring b
| 1304 |             self.p_engine_h = self.bench.ecm.engine_hours; | support logic; impact depends on neighboring bloc
| 1305 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1306 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1307 | (blank) | blank separator; no runtime behavior. |
| 1308 |     // ■ Update CAN signal map ■■■■■■■■■■■■■■■■
```

```
| 1330 |                 e.fresh = true; | support logic; impact depends on neighboring block and data flow. |
| 1331 |                 e.decoded = frame | support logic; impact depends on neighboring block and data flow. |
| 1332 |                     .decoded | support logic; impact depends on neighboring block and data flow. |
| 1333 |                     .iter() | support logic; impact depends on neighboring block and data flow. |
| 1334 |                     .map(\|v\| (v.name.to_string(), v.physical, v.unit.to_string())) | support logic; impact de
| 1335 |                     .collect(); | support logic; impact depends on neighboring block and data flow. |
| 1336 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1337 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1338 |         for sig in self.sig_map.values_mut() { | iteration logic; may alter timing, throughput, and safety cons
| 1339 |             if t - sig.last_ts > 0.25 { | runtime gate; condition changes can enable/disable critical paths. |
| 1340 |                 sig.fresh = false; | support logic; impact depends on neighboring block and data flow. |
| 1341 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1342 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1343 |         let trace_due = self | support logic; impact depends on neighboring block and data flow. |
| 1344 |             .can_trace_tick_counter | support logic; impact depends on neighboring block and data flow. |
| 1345 |             .is_multiple_of(self.can_trace_update_every_ticks.max(1)); | support logic; impact depends on neigh
| 1346 |         if self.can_trace_pause_ticks_left > 0 { | runtime gate; condition changes can enable/disable critical
| 1347 |             self.can_trace_pause_ticks_left = self.can_trace_pause_ticks_left.saturating_sub(1); | support logi
| 1348 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1349 |         if !self.can_freeze && trace_due && self.can_trace_pause_ticks_left == 0 { | runtime gate; condition ch
| 1350 |             self.trace_snap = self | support logic; impact depends on neighboring block and data flow. |
| 1351 |                 .bench | support logic; impact depends on neighboring block and data flow. |
| 1352 |                 .gateway | support logic; impact depends on neighboring block and data flow. |
| 1353 |                 .bus | support logic; impact depends on neighboring block and data flow. |
| 1354 |                 .frames | support logic; impact depends on neighboring block and data flow. |
| 1355 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 1356 |                 .map(\|f\| { | support logic; impact depends on neighboring block and data flow. |
| 1357 |                     ( | support logic; impact depends on neighboring block and data flow. |
| 1358 |                         f.timestamp, | support logic; impact depends on neighboring block and data flow. |
| 1359 |                         f.raw_id, | support logic; impact depends on neighboring block and data flow. |
| 1360 |                         f.sa, | support logic; impact depends on neighboring block and data flow. |
| 1361 |                         f.dlc, | support logic; impact depends on neighboring block and data flow. |
| 1362 |                         f.data_hex(), | support logic; impact depends on neighboring block and data flow. |
| 1363 |                         format!("{}[{}]", f.pgn_name(), f.sa_name()), | support logic; impact depends on neighb
| 1364 |                         f.decoded | support logic; impact depends on neighboring block and data flow. |
| 1365 |                             .iter() | support logic; impact depends on neighboring block and data flow. |
| 1366 |                             .take(3) | support logic; impact depends on neighboring block and data flow. |
| 1367 |                             .map(\|v\| format!("{:.1}{}", v.physical, v.unit)) | support logic; impact depends
| 1368 |                             .collect::<Vec<_>>() | support logic; impact depends on neighboring block and data
| 1369 |                             .join(" "), | support logic; impact depends on neighboring block and data flow. |
| 1370 |                     ) | support logic; impact depends on neighboring block and data flow. |
| 1371 |                 }) | support logic; impact depends on neighboring block and data flow. |
| 1372 |                 .collect(); | support logic; impact depends on neighboring block and data flow. |
| 1373 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1374 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1375 | (blank) | blank separator; no runtime behavior. |
| 1376 |     // ■ Generate meaningful events ■■■
```

```
| 1406 |                     if self.bench | runtime gate; condition changes can enable/disable critical paths. |
| 1407 |                         .boot | support logic; impact depends on neighboring block and data flow. |
| 1408 |                         .crank_inhibited { "INHIBITED" } else { "OK" } | support logic; impact depends on neigh
| 1409 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1410 |                 EventLevel::Info | support logic; impact depends on neighboring block and data flow. |
| 1411 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1412 |             self.prev_ign = ign; | support logic; impact depends on neighboring block and data flow. |
| 1413 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1414 |         // Engine start/stop | comment/documentation; changing affects understanding and intent traceability.
| 1415 |         if ecm.rpm > 400.0 && self.prev_rpm <= 400.0 { | runtime gate; condition changes can enable/disable cri
| 1416 |             ev!( | support logic; impact depends on neighboring block and data flow. |
| 1417 |                 "ECM", | support logic; impact depends on neighboring block and data flow. |
| 1418 |                 format!("ENGINE STARTED  RPM={:.0}  T={:.1}s", ecm.rpm, t), | support logic; impact depends on
| 1419 |                 EventLevel::Ok | support logic; impact depends on neighboring block and data flow. |
| 1420 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1421 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1422 |         if ecm.rpm <= 400.0 && self.prev_rpm > 400.0 { | runtime gate; condition changes can enable/disable cri
| 1423 |             ev!("ECM", "ENGINE STOPPED", EventLevel::Warn); | support logic; impact depends on neighboring bloc
| 1424 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1425 |         self.prev_rpm = ecm.rpm; | support logic; impact depends on neighboring block and data flow. |
| 1426 |         // Gear change | comment/documentation; changing affects understanding and intent traceability. |
| 1427 |         let gear = self.bench.tcm.gear_label.clone(); | support logic; impact depends on neighboring block and
| 1428 |         if gear != self.prev_gear { | runtime gate; condition changes can enable/disable critical paths. |
| 1429 |             ev!( | support logic; impact depends on neighboring block and data flow. |
| 1430 |                 "TCM", | support logic; impact depends on neighboring block and data flow. |
| 1431 |                 format!( | support logic; impact depends on neighboring block and data flow. |
| 1432 |                     "GEAR  {} → {}  (RPM {:.0}  Spd {:.1}km/h)", | support logic; impact depends on neighborin
| 1433 |                     self.prev_gear, gear, ecm.rpm, self.bench.tcm.ground_speed_kmh | support logic; impact dep
| 1434 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1435 |                 EventLevel::Info | support logic; impact depends on neighboring block and data flow. |
| 1436 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1437 |             self.prev_gear = gear; | support logic; impact depends on neighboring block and data flow. |
| 1438 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1439 |         // DTC set/clear | comment/documentation; changing affects understanding and intent traceability. |
| 1440 |         let dtcs = ecm.active_dtcs.len(); | support logic; impact depends on neighboring block and data flow.
| 1441 |         if dtcs > self.prev_dtcs { | runtime gate; condition changes can enable/disable critical paths. |
| 1442 |             if let Some(d) = ecm.active_dtcs.last() { | runtime gate; condition changes can enable/disable crit
| 1443 |                 let severity = format!("{}", d.severity); | support logic; impact depends on neighboring block
| 1444 |                 ev!( | support logic; impact depends on neighboring block and data flow. |
| 1445 |                     "ECM", | support logic; impact depends on neighboring block and data flow. |
| 1446 |                     format!( | support logic; impact depends on neighboring block and data flow. |
| 1447 |                         "DTC SET  SPN:{:5} FMI:{:2}  [{:4}]  {}", | support logic; impact depends on neighborin
| 1448 |                         d.spn, d.fmi, severity, d.desc | support logic; impact depends on neighboring block an
| 1449 |                     ), | support logic; impact depends on neighboring block and data flow. |
| 1450 |                     EventLevel::Critical | support logic; impact depends on neighboring block and data flow. |
| 1451 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 1452 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1453 |         } else if dtcs < self.prev_dtcs { | support logic; impact depends on neighboring block and data flow.
| 1454 |             ev!( | support logic; impact depends on neighboring block and data flow. |
| 1455 |                 "ECM", | support logic; impact depends on neighboring block and data flow. |
| 1456 |                 format!("DTCs CLEARED ({} were active)", self.prev_dtcs), | support logic; impact depends on ne
| 1457 |                 EventLevel::Ok | support logic; impact depends on neighboring block and data flow. |
| 1458 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1459 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1460 |         self.prev_dtcs = dtcs; | support logic; impact depends on neighboring block and data flow. |
| 1461 |         // Warning lamps | comment/documentation; changing affects understanding and intent traceability. |
| 1462 |         if ecm.red_lamp && self.prev_dtcs == 0 { | runtime gate; condition changes can enable/disable critical
| 1463 |             ev!("ECM", "■ RED STOP LAMP ON", EventLevel::Critical); | support logic; impact depends on neighbo
| 1464 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1465 |         if ecm.amber_lamp && t > 1.0 { /* only on transition */ } | runtime gate; condition changes can enable/
| 1466 |         // ABS | comment/documentation; changing affects understanding and intent traceability. |
| 1467 |         if abs.abs_system_active && !self.prev_abs { | runtime gate; condition changes can enable/disable crit
| 1468 |             ev!( | support logic; impact depends on neighboring block and data flow. |
| 1469 |                 "ABS", | support logic; impact depends on neighboring block and data flow. |
| 1470 |                 format!( | support logic; impact depends on neighboring block and data flow. |
| 1471 |                     "ABS ACTIVATED  Speed={:.1}km/h  Brake={:.0}%", | support logic; impact depends on neighbo
| 1472 |                     self.bench.tcm.ground_speed_kmh, self.bench.brake_pct | support logic; impact depends on ne
| 1473 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1474 |                 EventLevel::Warn | support logic; impact depends on neighboring block and data flow. |
| 1475 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1476 |         } else if !abs.abs_system_active && self.prev_abs { | support logic; impact depends on neighboring bloc
| 1477 |             ev!("ABS", "ABS DEACTIVATED — slip recovered", EventLevel::Ok); | support logic; impact depends on
| 1478 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1479 |         self.prev_abs = abs.abs_system_active; | support logic; impact depends on neighboring block and data f
| 1480 |         // ESP | comment/documentation; changing affects understanding and intent traceability. |
| 1481 |         if abs.esp_condition != self.prev_esp { | runtime gate; condition changes can enable/disable critical
```

```
| 1482 |             if abs.esp_condition != EspCondition::Neutral { | runtime gate; condition changes can enable/disabl
| 1483 |                 ev!( | support logic; impact depends on neighboring block and data flow. |
| 1484 |                     "ESP", | support logic; impact depends on neighboring block and data flow. |
| 1485 |                     format!( | support logic; impact depends on neighboring block and data flow. |
| 1486 |                         "ESP INTERVENTION: {:?}  Yaw-err={:+.1}°/s", | error propagation point; changes alter t
| 1487 |                         abs.esp_condition, abs.esp_yaw_error | support logic; impact depends on neighboring blo
| 1488 |                     ), | support logic; impact depends on neighboring block and data flow. |
| 1489 |                     EventLevel::Warn | support logic; impact depends on neighboring block and data flow. |
| 1490 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 1491 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 1492 |                 ev!("ESP", "ESP STABLE", EventLevel::Ok); | support logic; impact depends on neighboring block
| 1493 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1494 |             self.prev_esp = abs.esp_condition; | support logic; impact depends on neighboring block and data fl
| 1495 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1496 |         // DPF regen | comment/documentation; changing affects understanding and intent traceability. |
| 1497 |         let regen = self.bench.ecm.regen_requested; | support logic; impact depends on neighboring block and da
| 1498 |         if regen && !self.prev_regen { | runtime gate; condition changes can enable/disable critical paths. |
| 1499 |             ev!( | support logic; impact depends on neighboring block and data flow. |
| 1500 |                 "DPF", | support logic; impact depends on neighboring block and data flow. |
| 1501 |                 format!( | support logic; impact depends on neighboring block and data flow. |
| 1502 |                     "DPF REGEN STARTED  Soot={:.0}%  Exhaust={:.0}°C", | support logic; impact depends on neigh
| 1503 |                     ecm.dpf_soot_pct, ecm.exhaust_temp_c | support logic; impact depends on neighboring block a
| 1504 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1505 |                 EventLevel::Warn | support logic; impact depends on neighboring block and data flow. |
| 1506 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1507 |         } else if !regen && self.prev_regen { | support logic; impact depends on neighboring block and data flo
| 1508 |             ev!( | support logic; impact depends on neighboring block and data flow. |
| 1509 |                 "DPF", | support logic; impact depends on neighboring block and data flow. |
| 1510 |                 format!("DPF REGEN COMPLETE  Soot={:.0}%", ecm.dpf_soot_pct), | support logic; impact depends o
| 1511 |                 EventLevel::Ok | support logic; impact depends on neighboring block and data flow. |
| 1512 |             ); | support logic; impact depends on neighboring block and data flow. |
| 1513 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1514 |         self.prev_regen = regen; | support logic; impact depends on neighboring block and data flow. |
| 1515 |         // Coolant overheat | comment/documentation; changing affects understanding and intent traceability. |
| 1516 |         if ecm.coolant_temp_c > 102.0 && self.prev_rpm > 400.0 | runtime gate; condition changes can enable/dis
| 1517 |             && (t as u32).is_multiple_of(5) { | support logic; impact depends on neighboring block and data flo
| 1518 |                 // throttle event rate | comment/documentation; changing affects understanding and intent trace
| 1519 |                 ev!( | support logic; impact depends on neighboring block and data flow. |
| 1520 |                     "ECM", | support logic; impact depends on neighboring block and data flow. |
| 1521 |                     format!("HIGH COOLANT TEMP {:.1}°C", ecm.coolant_temp_c), | support logic; impact depends o
| 1522 |                     EventLevel::Critical | support logic; impact depends on neighboring block and data flow. |
| 1523 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 1524 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1525 |         // DEF warning | comment/documentation; changing affects understanding and intent traceability. |
| 1526 |         if ecm.def_level_pct < 10.0 && ecm.def_level_pct > 0.0 | runtime gate; condition changes can enable/dis
| 1527 |             && (t as u32).is_multiple_of(30) { | support logic; impact depends on neighboring block and data fl
| 1528 |                 ev!( | support logic; impact depends on neighboring block and data flow. |
| 1529 |                     "SCR", | support logic; impact depends on neighboring block and data flow. |
| 1530 |                     format!( | support logic; impact depends on neighboring block and data flow. |
| 1531 |                         "DEF LOW {:.1}%  NOx={:.0}ppm", | support logic; impact depends on neighboring block an
| 1532 |                         ecm.def_level_pct, ecm.nox_tailpipe_ppm | support logic; impact depends on neighboring
| 1533 |                     ), | support logic; impact depends on neighboring block and data flow. |
| 1534 |                     EventLevel::Warn | support logic; impact depends on neighboring block and data flow. |
| 1535 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 1536 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1537 |         // Boot events | comment/documentation; changing affects understanding and intent traceability. |
| 1538 |         let mode = format!("{}", self.bench.vcm.mode); | support logic; impact depends on neighboring block and
| 1539 |         if mode != self.prev_mode { | runtime gate; condition changes can enable/disable critical paths. |
| 1540 |             ev!("VCM", format!("Vehicle mode → {}", mode), EventLevel::Info); | support logic; impact depends
| 1541 |             self.prev_mode = mode; | support logic; impact depends on neighboring block and data flow. |
| 1542 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1543 |         // NM state | comment/documentation; changing affects understanding and intent traceability. |
| 1544 |         let nm_state = format!("{}", self.bench.net_mgmt.bus_state); | support logic; impact depends on neighbo
| 1545 |         let _ = nm_state; // could add NM events | support logic; impact depends on neighboring block and data
| 1546 |                           // TCS | comment/documentation; changing affects understanding and intent traceabili
| 1547 |         if abs.tcs_system_active | runtime gate; condition changes can enable/disable critical paths. |
| 1548 |             && (t as u32).is_multiple_of(2) { | support logic; impact depends on neighboring block and data flo
| 1549 |                 ev!( | support logic; impact depends on neighboring block and data flow. |
| 1550 |                     "TCS", | support logic; impact depends on neighboring block and data flow. |
| 1551 |                     format!( | support logic; impact depends on neighboring block and data flow. |
| 1552 |                         "TCS cut={:.0}%  Speed={:.1}km/h", | support logic; impact depends on neighboring bloc
| 1553 |                         abs.tcs_throttle_cut * 100.0, | support logic; impact depends on neighboring block and
| 1554 |                         self.bench.tcm.ground_speed_kmh | support logic; impact depends on neighboring block an
| 1555 |                     ), | support logic; impact depends on neighboring block and data flow. |
| 1556 |                     EventLevel::Warn | support logic; impact depends on neighboring block and data flow. |
| 1557 |                 ); | support logic; impact depends on neighboring block and data flow. |
```

```
| 1558 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1559 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1560 | (blank) | blank separator; no runtime behavior. |
| 1561 |     fn push_plots(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1562 |         let t = self.bench.elapsed; | support logic; impact depends on neighboring block and data flow. |
| 1563 |         macro_rules! p { | support logic; impact depends on neighboring block and data flow. |
| 1564 |             ($v:expr,$x:expr) => { | support logic; impact depends on neighboring block and data flow. |
| 1565 |                 $v.push([t, $x]); | support logic; impact depends on neighboring block and data flow. |
| 1566 |                 if $v.len() > PLOT_WINDOW { | runtime gate; condition changes can enable/disable critical paths
| 1567 |                     $v.remove(0); | support logic; impact depends on neighboring block and data flow. |
| 1568 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1569 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1570 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1571 |         p!(self.pl_rpm, self.bench.ecm.rpm); | support logic; impact depends on neighboring block and data flow
| 1572 |         p!(self.pl_spd, self.bench.tcm.ground_speed_kmh); | support logic; impact depends on neighboring block
| 1573 |         p!(self.pl_torque, self.bench.ecm.actual_torque_nm); | support logic; impact depends on neighboring bl
| 1574 |         p!(self.pl_fuel, self.bench.ecm.fuel_rate_lph); | support logic; impact depends on neighboring block a
| 1575 |         p!(self.pl_coolant, self.bench.ecm.coolant_temp_c); | support logic; impact depends on neighboring bloc
| 1576 |         p!(self.pl_dpf, self.bench.ecm.dpf_soot_pct); | support logic; impact depends on neighboring block and
| 1577 |         p!(self.pl_boost, self.bench.ecm.boost_pressure_kpa); | support logic; impact depends on neighboring b
| 1578 |         p!(self.pl_hyd, self.bench.hcm.system_pressure_bar); | support logic; impact depends on neighboring bl
| 1579 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1580 | (blank) | blank separator; no runtime behavior. |
| 1581 |     fn auto_sequence(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 1582 |         match self.ticks { | branch dispatcher; arm changes alter behavior class handling. |
| 1583 |             60 => self.cmds.push(Cmd::KeyAdvance), | support logic; impact depends on neighboring block and da
| 1584 |             120 => self.cmds.push(Cmd::KeyAdvance), | support logic; impact depends on neighboring block and d
| 1585 |             180 => self.cmds.push(Cmd::KeyAdvance), | support logic; impact depends on neighboring block and d
| 1586 |             _ => {} | support logic; impact depends on neighboring block and data flow. |
| 1587 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1588 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1589 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1590 | (blank) | blank separator; no runtime behavior. |
| 1591 | fn now_ms() -> u64 { | function signature/body; changes affect API behavior and control-flow. |
| 1592 |     std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 1593 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data f
| 1594 |         .map(\|d\| d.as_millis() as u64) | support logic; impact depends on neighboring block and data flow. |
| 1595 |         .unwrap_or(0) | support logic; impact depends on neighboring block and data flow. |
| 1596 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1597 | (blank) | blank separator; no runtime behavior. |
| 1598 | // ■■■
```

```
| 1634 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1635 | (blank) | blank separator; no runtime behavior. |
| 1636 |     eframe::run_native( | support logic; impact depends on neighboring block and data flow. |
| 1637 |         "■ Heavy Machinery ECU Bench v3.0", | support logic; impact depends on neighboring block and data flow
| 1638 |         NativeOptions { | support logic; impact depends on neighboring block and data flow. |
| 1639 |             viewport: ViewportBuilder::default() | support logic; impact depends on neighboring block and data
| 1640 |                 .with_inner_size([1700.0, 1020.0]) | support logic; impact depends on neighboring block and da
| 1641 |                 .with_min_inner_size([1200.0, 700.0]), | support logic; impact depends on neighboring block an
| 1642 |             ..Default::default() | support logic; impact depends on neighboring block and data flow. |
| 1643 |         }, | support logic; impact depends on neighboring block and data flow. |
| 1644 |         Box::new(move \|cc\| Ok(Box::new(App::new(cc, hw_cfg.clone())))), | support logic; impact depends on ne
| 1645 |     ) | support logic; impact depends on neighboring block and data flow. |
| 1646 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1647 | (blank) | blank separator; no runtime behavior. |
| 1648 | // ■■■
```

```
| 1710 |             if !feed.is_alive() { | runtime gate; condition changes can enable/disable critical paths. |
| 1711 |                 if let Some(err) = feed.last_error() { | runtime gate; condition changes can enable/disable cri
| 1712 |                     self.ecm_live_status = format!("Retrieve stopped with error: {}", err); | support logic; im
| 1713 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 1714 |                     self.ecm_live_status = "Retrieve stopped.".into(); | support logic; impact depends on neigh
| 1715 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1716 |                 clear_live_feed = true; | support logic; impact depends on neighboring block and data flow. |
| 1717 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1718 |         } else if matches!(self.hw_cfg.mode, io::hw::HwMode::Sim) { | support logic; impact depends on neighbo
| 1719 |             self.refresh_mock_ecm_live_from_sim(); | support logic; impact depends on neighboring block and da
| 1720 |             self.ecm_live_frames_total = self.ecm_live_frames_total.saturating_add(1); | support logic; impact
| 1721 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1722 |         if clear_live_feed { | runtime gate; condition changes can enable/disable critical paths. |
| 1723 |             self.ecm_live_feed = None; | support logic; impact depends on neighboring block and data flow. |
| 1724 |             self.ecm_connected = false; | support logic; impact depends on neighboring block and data flow. |
| 1725 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1726 |         ctx.request_repaint(); | support logic; impact depends on neighboring block and data flow. |
| 1727 | (blank) | blank separator; no runtime behavior. |
| 1728 |         // ■■ Panels ■■■
```

```
| 1786 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1787 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1788 |                     "DIAGNOSTICO", | support logic; impact depends on neighboring block and data flow. |
| 1789 |                     &[ | support logic; impact depends on neighboring block and data flow. |
| 1790 |                         (Tab::CanBus, "■ CAN"), | support logic; impact depends on neighboring block and data
| 1791 |                         (Tab::Events, "■ EVENTS"), | support logic; impact depends on neighboring block and da
| 1792 |                         (Tab::EcuNet, "■ ECU NET"), | support logic; impact depends on neighboring block and d
| 1793 |                         (Tab::Faults, "■ FAULTS"), | support logic; impact depends on neighboring block and da
| 1794 |                         (Tab::Boot, "■ BOOT"), | support logic; impact depends on neighboring block and data f
| 1795 |                         (Tab::Params, "■ PARAMS"), | support logic; impact depends on neighboring block and da
| 1796 |                         (Tab::Sensors, "■ SENSORS"), | support logic; impact depends on neighboring block and
| 1797 |                         (Tab::V2X, "■ V2X"), | support logic; impact depends on neighboring block and data flo
| 1798 |                         (Tab::Uds, "■ UDS"), | protocol or I/O critical; changes may impact external integrati
| 1799 |                         (Tab::EcmLiveData, "■ ECM LIVE"), | support logic; impact depends on neighboring block
| 1800 |                     ], | support logic; impact depends on neighboring block and data flow. |
| 1801 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1802 |             ]; | support logic; impact depends on neighboring block and data flow. |
| 1803 | (blank) | blank separator; no runtime behavior. |
| 1804 |             for (section_name, tabs) in nav_sections { | iteration logic; may alter timing, throughput, and sa
| 1805 |                 ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data f
| 1806 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 1807 |                         RichText::new(section_name) | support logic; impact depends on neighboring block and da
| 1808 |                             .size(9.6) | support logic; impact depends on neighboring block and data flow. |
| 1809 |                             .color(Color32::from_gray(145)) | support logic; impact depends on neighboring bloc
| 1810 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 1811 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 1812 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 1813 |                     for (t, lbl) in tabs { | iteration logic; may alter timing, throughput, and safety constra
| 1814 |                         let sel = self.tab == *t; | support logic; impact depends on neighboring block and data
| 1815 |                         let fill = if sel { | support logic; impact depends on neighboring block and data flow
| 1816 |                             Color32::from_rgb(45, 105, 210) | support logic; impact depends on neighboring bloc
| 1817 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 1818 |                             Color32::from_gray(30) | support logic; impact depends on neighboring block and da
| 1819 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 1820 |                         let label = if *t == Tab::Faults && dtcs > 0 { | support logic; impact depends on neig
| 1821 |                             format!("■ FAULTS({})", dtcs) | support logic; impact depends on neighboring block
| 1822 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 1823 |                             (*lbl).into() | support logic; impact depends on neighboring block and data flow.
| 1824 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 1825 |                         let col = if *t == Tab::Faults && dtcs > 0 { | support logic; impact depends on neighb
| 1826 |                             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 1827 |                         } else if sel { | support logic; impact depends on neighboring block and data flow. |
| 1828 |                             Color32::WHITE | support logic; impact depends on neighboring block and data flow.
| 1829 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 1830 |                             Color32::from_gray(160) | support logic; impact depends on neighboring block and da
| 1831 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 1832 |                         if ui | runtime gate; condition changes can enable/disable critical paths. |
| 1833 |                             .add(Button::new(RichText::new(label).size(10.8).color(col)).fill(fill)) | support
| 1834 |                             .clicked() | support logic; impact depends on neighboring block and data flow. |
| 1835 |                         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1836 |                             self.tab = *t; | support logic; impact depends on neighboring block and data flow.
| 1837 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 1838 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1839 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 1840 |                 if section_name == "OPERACAO" { | runtime gate; condition changes can enable/disable critical
| 1841 |                     ui.add_space(2.0); | support logic; impact depends on neighboring block and data flow. |
| 1842 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1843 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1844 |         }); | support logic; impact depends on neighboring block and data flow. |
| 1845 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1846 | (blank) | blank separator; no runtime behavior. |
| 1847 |     fn ad_readiness_issues(&self) -> Vec<String> { | function signature/body; changes affect API behavior and
| 1848 |         let mut issues = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 1849 |         if !self.bench.engine_running() { | runtime gate; condition changes can enable/disable critical paths.
| 1850 |             issues.push("Motor desligado: o AD nao consegue acelerar nem sustentar velocidade.".to_string());
| 1851 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1852 |         if matches!(self.bench.tcm.direction, Direction::Park) { | runtime gate; condition changes can enable/
| 1853 |             issues.push("Transmissao em Park: selecione Forward para permitir deslocamento.".to_string()); | s
| 1854 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1855 |         if self.bench.tcm.auto_mode != AutoShiftMode::Auto { | runtime gate; condition changes can enable/disa
| 1856 |             issues.push("Transmissao fora de AUTO: o AD perde previsibilidade de troca.".to_string()); | suppo
| 1857 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1858 |         if self.bench.ad.lane.confidence <= 0.4 { | runtime gate; condition changes can enable/disable critica
| 1859 |             issues.push("Lane confidence baixa: o LKA permanece em espera abaixo de 40%.".to_string()); | supp
| 1860 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1861 |         if self.bench.tcm.ground_speed_kmh < 5.0 { | runtime gate; condition changes can enable/disable critica
```

```
| 1862 |             issues.push("Abaixo de 5 km/h o LKA ainda nao atua; o ACC pode acelerar a maquina.".to_string());
| 1863 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1864 |         issues | support logic; impact depends on neighboring block and data flow. |
| 1865 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1866 | (blank) | blank separator; no runtime behavior. |
| 1867 |     fn ux_alerts(&self) -> Vec<(Color32, String)> { | function signature/body; changes affect API behavior and
| 1868 |         let mut alerts = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 1869 |         match self.tab { | branch dispatcher; arm changes alter behavior class handling. |
| 1870 |             Tab::Cluster \| Tab::Electrical \| Tab::Engine \| Tab::Implements => { | support logic; impact depe
| 1871 |                 if !self.bench.engine_running() { | runtime gate; condition changes can enable/disable critical
| 1872 |                     alerts.push(( | support logic; impact depends on neighboring block and data flow. |
| 1873 |                         Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 1874 |                         "Motor desligado: esta aba so reflete dinamica real depois de RUN.".to_string(), | supp
| 1875 |                     )); | support logic; impact depends on neighboring block and data flow. |
| 1876 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1877 |                 if self.bench.ecm.red_lamp { | runtime gate; condition changes can enable/disable critical path
| 1878 |                     alerts.push(( | support logic; impact depends on neighboring block and data flow. |
| 1879 |                         Color32::RED, | support logic; impact depends on neighboring block and data flow. |
| 1880 |                         "RED STOP ativo: existe falha critica impactando a operacao.".to_string(), | support lo
| 1881 |                     )); | support logic; impact depends on neighboring block and data flow. |
| 1882 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1883 |                 if matches!(self.tab, Tab::Electrical) | runtime gate; condition changes can enable/disable cri
| 1884 |                     && self.bench.bcm.fuses.iter().any(\|f\| f.blown) | support logic; impact depends on neighb
| 1885 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1886 |                     alerts.push(( | support logic; impact depends on neighboring block and data flow. |
| 1887 |                         Color32::RED, | support logic; impact depends on neighboring block and data flow. |
| 1888 |                         "Existe ao menos um fusivel aberto: rearme antes de validar o ramo afetado.".to_string
| 1889 |                     )); | support logic; impact depends on neighboring block and data flow. |
| 1890 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1891 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1892 |             Tab::Autonomous => { | support logic; impact depends on neighboring block and data flow. |
| 1893 |                 if !self.bench.engine_running() { | runtime gate; condition changes can enable/disable critical
| 1894 |                     alerts.push(( | support logic; impact depends on neighboring block and data flow. |
| 1895 |                         Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 1896 |                         "Motor desligado: o AD nao consegue gerar tracao nem validar longitudinal.".to_string(
| 1897 |                     )); | support logic; impact depends on neighboring block and data flow. |
| 1898 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1899 |                 if self.bench.ad.engaged { | runtime gate; condition changes can enable/disable critical paths
| 1900 |                     let msg = if let Some(reason) = self.bench.ad.degrade_reason { | support logic; impact depe
| 1901 |                         format!("AD engatado em modo degradado: {}.", reason) | support logic; impact depends o
| 1902 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 1903 |                         "AD engatado: comandos manuais principais ficam travados para evitar disputa.".to_strin
| 1904 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1905 |                     alerts.push((Color32::from_rgb(90, 210, 130), msg)); | support logic; impact depends on nei
| 1906 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1907 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1908 |             Tab::CanBus \| Tab::Events \| Tab::EcuNet \| Tab::Faults \| Tab::Uds => { | protocol or I/O critica
| 1909 |                 if self.bench.gateway.bus_load_pct > 85.0 { | runtime gate; condition changes can enable/disabl
| 1910 |                     alerts.push(( | support logic; impact depends on neighboring block and data flow. |
| 1911 |                         Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 1912 |                         format!("Carga CAN alta ({:.0}%): diagnostico pode ficar ruidoso nesta aba.", self.bench
| 1913 |                     )); | support logic; impact depends on neighboring block and data flow. |
| 1914 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1915 |                 if matches!(self.tab, Tab::Faults \| Tab::Uds) && self.bench.ecm.red_lamp { | runtime gate; con
| 1916 |                     alerts.push(( | support logic; impact depends on neighboring block and data flow. |
| 1917 |                         Color32::RED, | support logic; impact depends on neighboring block and data flow. |
| 1918 |                         "Falha critica ativa: compare Faults, Events e Engine antes de continuar.".to_string(),
| 1919 |                     )); | support logic; impact depends on neighboring block and data flow. |
| 1920 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1921 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1922 |             Tab::EcmLiveData => { | support logic; impact depends on neighboring block and data flow. |
| 1923 |                 if self.ecm_live_feed.as_ref().is_some_and(\|feed\| !feed.is_alive()) { | runtime gate; conditi
| 1924 |                     alerts.push(( | support logic; impact depends on neighboring block and data flow. |
| 1925 |                         Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 1926 |                         "ECM Live sem stream valido: reconecte ou reinicie Retrieve Data.".to_string(), | suppo
| 1927 |                     )); | support logic; impact depends on neighboring block and data flow. |
| 1928 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1929 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1930 |             Tab::Help \| Tab::Boot \| Tab::Params \| Tab::Sensors \| Tab::V2X \| Tab::LeakLab \| Tab::Plots =>
| 1931 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1932 |         alerts | support logic; impact depends on neighboring block and data flow. |
| 1933 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 1934 | (blank) | blank separator; no runtime behavior. |
| 1935 |     fn ux_context_items(&self) -> Vec<(String, Color32)> { | function signature/body; changes affect API behavi
| 1936 |         match self.tab { | branch dispatcher; arm changes alter behavior class handling. |
| 1937 |             Tab::Help => vec![ | support logic; impact depends on neighboring block and data flow. |
```

```
| 1938 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1939 |                     format!("Workspace {:.1}s", self.bench.elapsed), | support logic; impact depends on neighbo
| 1940 |                     Color32::from_gray(180), | support logic; impact depends on neighboring block and data flow
| 1941 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1942 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1943 |                     format!("{} abas operacionais", 17), | support logic; impact depends on neighboring block a
| 1944 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 1945 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1946 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1947 |                     format!("{} DTC ativos", self.bench.ecm.active_dtcs.len()), | support logic; impact depends
| 1948 |                     if self.bench.ecm.active_dtcs.is_empty() { | runtime gate; condition changes can enable/dis
| 1949 |                         Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 1950 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 1951 |                         Color32::YELLOW | support logic; impact depends on neighboring block and data flow. |
| 1952 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 1953 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1954 |             ], | support logic; impact depends on neighboring block and data flow. |
| 1955 |             Tab::Cluster => vec![ | support logic; impact depends on neighboring block and data flow. |
| 1956 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1957 |                     format!("RPM {:.0}", self.bench.ecm.rpm), | support logic; impact depends on neighboring bl
| 1958 |                     Color32::GREEN, | support logic; impact depends on neighboring block and data flow. |
| 1959 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1960 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1961 |                     format!("SPD {:.1} km/h", self.bench.tcm.ground_speed_kmh), | support logic; impact depends
| 1962 |                     Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and data flow. |
| 1963 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1964 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1965 |                     format!("Gear {}", self.bench.tcm.gear_label), | support logic; impact depends on neighbori
| 1966 |                     Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 1967 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1968 |             ], | support logic; impact depends on neighboring block and data flow. |
| 1969 |             Tab::Electrical => vec![ | support logic; impact depends on neighboring block and data flow. |
| 1970 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1971 |                     format!("Batt {:.1} V", self.bench.bcm.battery_voltage), | support logic; impact depends on
| 1972 |                     if self.bench.bcm.battery_voltage < 11.8 { | runtime gate; condition changes can enable/dis
| 1973 |                         Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 1974 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 1975 |                         Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 1976 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 1977 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1978 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1979 |                     format!("Load {:.0} A", self.bench.bcm.total_load_amps), | support logic; impact depends on
| 1980 |                     Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 1981 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1982 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1983 |                     format!("Load shed {}", if self.bench.vcm.power.load_shed_active { "on" } else { "off" }),
| 1984 |                     if self.bench.vcm.power.load_shed_active { | runtime gate; condition changes can enable/dis
| 1985 |                         Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 1986 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 1987 |                         Color32::from_gray(180) | support logic; impact depends on neighboring block and data f
| 1988 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 1989 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 1990 |             ], | support logic; impact depends on neighboring block and data flow. |
| 1991 |             Tab::CanBus => vec![ | support logic; impact depends on neighboring block and data flow. |
| 1992 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 1993 |                     format!("Bus load {:.0}%", self.bench.gateway.bus_load_pct), | support logic; impact depend
| 1994 |                     if self.bench.gateway.bus_load_pct > 85.0 { | runtime gate; condition changes can enable/di
| 1995 |                         Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 1996 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 1997 |                         Color32::GOLD | support logic; impact depends on neighboring block and data flow. |
| 1998 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 1999 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2000 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2001 |                     format!("Signals {}", self.sig_map.len()), | support logic; impact depends on neighboring b
| 2002 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2003 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2004 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2005 |                     format!("Trace {}", self.trace_snap.len()), | support logic; impact depends on neighboring
| 2006 |                     Color32::from_gray(180), | support logic; impact depends on neighboring block and data flow
| 2007 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2008 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2009 |             Tab::Events => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2010 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2011 |                     format!("Eventos {}", self.events.len()), | support logic; impact depends on neighboring bl
| 2012 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2013 |                 ), | support logic; impact depends on neighboring block and data flow. |
```

```
| 2014 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2015 |                     format!("Filtro {}", if self.ev_filter.is_empty() { "livre" } else { "ativo" }), | support
| 2016 |                     if self.ev_filter.is_empty() { | runtime gate; condition changes can enable/disable critica
| 2017 |                         Color32::from_gray(180) | support logic; impact depends on neighboring block and data f
| 2018 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2019 |                         Color32::YELLOW | support logic; impact depends on neighboring block and data flow. |
| 2020 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 2021 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2022 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2023 |                     format!("Pause {}", if self.ev_pause { "on" } else { "off" }), | support logic; impact depe
| 2024 |                     if self.ev_pause { Color32::YELLOW } else { Color32::GREEN }, | runtime gate; condition cha
| 2025 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2026 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2027 |             Tab::EcuNet => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2028 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2029 |                     format!( | support logic; impact depends on neighboring block and data flow. |
| 2030 |                         "Online {}/{}", | support logic; impact depends on neighboring block and data flow. |
| 2031 |                         self.bench.boot.ecus.iter().filter(\|e\| e.is_online()).count(), | support logic; impac
| 2032 |                         self.bench.boot.ecus.len() | support logic; impact depends on neighboring block and da
| 2033 |                     ), | support logic; impact depends on neighboring block and data flow. |
| 2034 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2035 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2036 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2037 |                     format!("NM {}", self.bench.net_mgmt.bus_state), | support logic; impact depends on neighbo
| 2038 |                     Color32::from_gray(180), | support logic; impact depends on neighboring block and data flow
| 2039 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2040 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2041 |             Tab::Engine => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2042 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2043 |                     format!("Load {:.0}%", self.bench.ecm.percent_load), | support logic; impact depends on nei
| 2044 |                     Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 2045 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2046 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2047 |                     format!("Torque {:.0} Nm", self.bench.ecm.actual_torque_nm), | support logic; impact depend
| 2048 |                     Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighboring block and d
| 2049 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2050 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2051 |                     format!("Coolant {:.0} C", self.bench.ecm.coolant_temp_c), | support logic; impact depends
| 2052 |                     if self.bench.ecm.coolant_temp_c > 105.0 { | runtime gate; condition changes can enable/dis
| 2053 |                         Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 2054 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2055 |                         Color32::from_rgb(255, 130, 30) | support logic; impact depends on neighboring block an
| 2056 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 2057 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2058 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2059 |             Tab::Faults => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2060 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2061 |                     format!("Active DTC {}", self.bench.ecm.active_dtcs.len()), | support logic; impact depends
| 2062 |                     if self.bench.ecm.active_dtcs.is_empty() { | runtime gate; condition changes can enable/dis
| 2063 |                         Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 2064 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2065 |                         Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 2066 |                     }, | support logic; impact depends on neighboring block and data flow. |
| 2067 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2068 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2069 |                     format!("Selected {}", self.bench.selected_fault), | support logic; impact depends on neigh
| 2070 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2071 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2072 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2073 |                     format!("Fault inject {}", if self.bench.fault_active { "armed" } else { "idle" }), | suppo
| 2074 |                     if self.bench.fault_active { Color32::YELLOW } else { Color32::GREEN }, | runtime gate; con
| 2075 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2076 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2077 |             Tab::Boot => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2078 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2079 |                     format!("Ignition {}", self.bench.ignition()), | support logic; impact depends on neighbori
| 2080 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2081 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2082 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2083 |                     format!("Engine {}", if self.bench.engine_running() { "RUN" } else { "OFF" }), | support lo
| 2084 |                     if self.bench.engine_running() { Color32::GREEN } else { Color32::YELLOW }, | runtime gate
| 2085 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2086 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2087 |             Tab::Implements => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2088 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2089 |                     format!("PTO {}", if self.bench.implement.pto_rear_enabled { "on" } else { "off" }), | supp
```

```
| 2090 |                     if self.bench.implement.pto_rear_enabled { Color32::GREEN } else { Color32::from_gray(170)
| 2091 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2092 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2093 |                     format!("Hyd {:.0} bar", self.bench.hcm.system_pressure_bar), | support logic; impact depen
| 2094 |                     Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring block and da
| 2095 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2096 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2097 |             Tab::Params => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2098 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2099 |                     "Live tuning panel".to_string(), | support logic; impact depends on neighboring block and d
| 2100 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2101 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2102 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2103 |                     format!("Mode {}", if self.ux_compact_mode { "compact" } else { "full" }), | support logic
| 2104 |                     Color32::from_gray(180), | support logic; impact depends on neighboring block and data flow
| 2105 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2106 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2107 |             Tab::Sensors => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2108 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2109 |                     format!("GPS {:.1} km/h", self.bench.gps.speed_kmh), | support logic; impact depends on nei
| 2110 |                     Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and data flow. |
| 2111 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2112 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2113 |                     format!("Radar TTC {:.1}s", self.bench.radar.ttc_front.min(99.9)), | support logic; impact
| 2114 |                     if self.bench.radar.ttc_front < 2.0 { Color32::RED } else { Color32::YELLOW }, | runtime ga
| 2115 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2116 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2117 |                     format!("Lane conf {:.0}%", self.bench.ad.lane.confidence * 100.0), | support logic; impact
| 2118 |                     if self.bench.ad.lane.confidence > 0.7 { Color32::GREEN } else { Color32::YELLOW }, | runti
| 2119 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2120 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2121 |             Tab::Autonomous => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2122 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2123 |                     format!("AD {}", if self.bench.ad.engaged { "engaged" } else { "standby" }), | support logi
| 2124 |                     if self.bench.ad.engaged { Color32::GREEN } else { Color32::from_gray(180) }, | runtime ga
| 2125 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2126 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2127 |                     format!("ACC set {:.0} km/h", self.bench.ad.acc_set_speed_kmh), | support logic; impact dep
| 2128 |                     Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and data flow. |
| 2129 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2130 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2131 |                     format!("Lane {:.0}%", self.bench.ad.lane.confidence * 100.0), | support logic; impact depe
| 2132 |                     if self.bench.ad.lane.confidence > 0.7 { Color32::GREEN } else { Color32::YELLOW }, | runti
| 2133 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2134 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2135 |             Tab::V2X => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2136 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2137 |                     format!("Link {:.0}%", self.v2x_range_ema.mul_add(0.0, (100.0 - (self.v2x_loss_ema * 2.0).
| 2138 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2139 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2140 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2141 |                     format!("Nearby {}", self.bench.v2x.nearby_vehicles.len()), | support logic; impact depends
| 2142 |                     Color32::from_gray(180), | support logic; impact depends on neighboring block and data flow
| 2143 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2144 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2145 |                     format!("OTA {}", if self.bench.telematics.ota_downloading { "downloading" } else if self.b
| 2146 |                     if self.bench.telematics.ota_downloading { Color32::YELLOW } else { Color32::GREEN }, | run
| 2147 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2148 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2149 |             Tab::Uds => vec![ | protocol or I/O critical; changes may impact external integration correctness.
| 2150 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2151 |                     format!("Session {}", self.bench.uds_ecm.session), | protocol or I/O critical; changes may
| 2152 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2153 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2154 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2155 |                     format!("Security {:?}", self.bench.uds_ecm.security), | error propagation point; changes a
| 2156 |                     Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 2157 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2158 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2159 |                     format!("UDS log {}", self.uds_log.len()), | protocol or I/O critical; changes may impact c
| 2160 |                     Color32::from_gray(180), | support logic; impact depends on neighboring block and data flow
| 2161 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2162 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2163 |             Tab::EcmLiveData => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2164 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2165 |                     format!("Mode {:?}", self.hw_cfg.mode), | error propagation point; changes alter failure pr
```

```
| 2166 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2167 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2168 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2169 |                     format!("Detected {}", self.ecm_detected_sas.len()), | support logic; impact depends on nei
| 2170 |                     Color32::from_gray(180), | support logic; impact depends on neighboring block and data flo
| 2171 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2172 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2173 |                     format!("Feed {}", if self.ecm_live_feed.is_some() { "active" } else { "idle" }), | suppor
| 2174 |                     if self.ecm_live_feed.is_some() { Color32::GREEN } else { Color32::YELLOW }, | runtime gate
| 2175 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2176 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2177 |             Tab::LeakLab => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2178 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2179 |                     format!("Circuitos {}", self.bench.leak_rig.circuits.len()), | support logic; impact depend
| 2180 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2181 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2182 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2183 |                     format!("Predicoes {}", self.leak_predictions.len()), | support logic; impact depends on ne
| 2184 |                     Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 2185 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2186 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2187 |                     format!("Calibrado {}", if self.leak_calibration_report.is_some() { "sim" } else { "nao" }
| 2188 |                     if self.leak_calibration_report.is_some() { Color32::GREEN } else { Color32::from_gray(180
| 2189 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2190 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2191 |             Tab::Plots => vec![ | support logic; impact depends on neighboring block and data flow. |
| 2192 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2193 |                     format!("Samples {}", self.pl_rpm.len()), | support logic; impact depends on neighboring b
| 2194 |                     Color32::from_rgb(110, 180, 255), | support logic; impact depends on neighboring block and
| 2195 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2196 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 2197 |                     format!("SPD {:.1} km/h", self.bench.tcm.ground_speed_kmh), | support logic; impact depends
| 2198 |                     Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and data flow. |
| 2199 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 2200 |             ], | support logic; impact depends on neighboring block and data flow. |
| 2201 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2202 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2203 | (blank) | blank separator; no runtime behavior. |
| 2204 |     fn render_ux_context_strip(&self, ui: &mut Ui) { | function signature/body; changes affect API behavior and
| 2205 |         let items = self.ux_context_items(); | support logic; impact depends on neighboring block and data flo
| 2206 |         if items.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 2207 |             return; | support logic; impact depends on neighboring block and data flow. |
| 2208 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2209 |         ui.add_space(4.0); | support logic; impact depends on neighboring block and data flow. |
| 2210 |         ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2211 |             for (text, color) in items { | iteration logic; may alter timing, throughput, and safety constrain
| 2212 |                 egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data f
| 2213 |                     .fill(Color32::from_gray(20)) | support logic; impact depends on neighboring block and data
| 2214 |                     .inner_margin(Margin::symmetric(8.0, 4.0)) | support logic; impact depends on neighboring b
| 2215 |                     .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2216 |                         ui.label(RichText::new(text).size(10.0).color(color)); | support logic; impact depends
| 2217 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2218 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2219 |         }); | support logic; impact depends on neighboring block and data flow. |
| 2220 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2221 | (blank) | blank separator; no runtime behavior. |
| 2222 |     fn statusbar(&self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flow
| 2223 |         let t = self.bench.elapsed; | support logic; impact depends on neighboring block and data flow. |
| 2224 |         let rpm = self.bench.ecm.rpm; | support logic; impact depends on neighboring block and data flow. |
| 2225 |         let spd = self.bench.tcm.ground_speed_kmh; | support logic; impact depends on neighboring block and da
| 2226 |         let gear = &self.bench.tcm.gear_label; | support logic; impact depends on neighboring block and data f
| 2227 |         let can = self.bench.gateway.bus_load_pct; | support logic; impact depends on neighboring block and da
| 2228 |         let dtcs = self.bench.ecm.active_dtcs.len(); | support logic; impact depends on neighboring block and
| 2229 |         let nm = format!("{}", self.bench.net_mgmt.bus_state); | support logic; impact depends on neighboring b
| 2230 |         let red = self.bench.ecm.red_lamp; | support logic; impact depends on neighboring block and data flow.
| 2231 |         let amber = self.bench.ecm.amber_lamp; | support logic; impact depends on neighboring block and data f
| 2232 |         let abs_on = self.bench.abs.abs_system_active; | support logic; impact depends on neighboring block an
| 2233 |         let esp_on = self.bench.abs.esp_system_active; | support logic; impact depends on neighboring block an
| 2234 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2235 |             macro_rules! s { | support logic; impact depends on neighboring block and data flow. |
| 2236 |                 ($v:expr,$c:expr) => { | support logic; impact depends on neighboring block and data flow. |
| 2237 |                     ui.label(RichText::new($v).size(11.0).color($c)); | support logic; impact depends on neigh
| 2238 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2239 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2240 |             s!(format!("t={:.1}s", t), Color32::from_gray(120)); | support logic; impact depends on neighborin
| 2241 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
```

```
| 2242 |             s!(format!("RPM {:.0}", rpm), Color32::GREEN); | support logic; impact depends on neighboring bloc
| 2243 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2244 |             s!(format!("{:.1}km/h", spd), Color32::LIGHT_BLUE); | support logic; impact depends on neighboring
| 2245 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2246 |             s!(format!("Gear {}", gear), Color32::YELLOW); | support logic; impact depends on neighboring bloc
| 2247 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2248 |             s!(format!("CAN {:.0}%", can), Color32::GOLD); | support logic; impact depends on neighboring bloc
| 2249 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2250 |             s!( | support logic; impact depends on neighboring block and data flow. |
| 2251 |                 format!("{} DTC", dtcs), | support logic; impact depends on neighboring block and data flow. |
| 2252 |                 if dtcs == 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 2253 |                     Color32::from_gray(100) | support logic; impact depends on neighboring block and data flow
| 2254 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 2255 |                     Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 2256 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2257 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2258 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2259 |             s!(format!("NM:{}", nm), Color32::from_gray(120)); | support logic; impact depends on neighboring
| 2260 |             if red { | runtime gate; condition changes can enable/disable critical paths. |
| 2261 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2262 |                 s!("■ RED STOP", Color32::RED); | support logic; impact depends on neighboring block and data
| 2263 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2264 |             if amber { | runtime gate; condition changes can enable/disable critical paths. |
| 2265 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2266 |                 s!("■ AMBER", Color32::GOLD); | support logic; impact depends on neighboring block and data fl
| 2267 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2268 |             if abs_on { | runtime gate; condition changes can enable/disable critical paths. |
| 2269 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2270 |                 s!("■ ABS", Color32::WHITE); | support logic; impact depends on neighboring block and data flo
| 2271 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2272 |             if esp_on { | runtime gate; condition changes can enable/disable critical paths. |
| 2273 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2274 |                 s!("■ ESP", Color32::LIGHT_BLUE); | support logic; impact depends on neighboring block and dat
| 2275 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2276 |             if self.bench.ad.engaged { | runtime gate; condition changes can enable/disable critical paths. |
| 2277 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2278 |                 s!("■ AD ON", Color32::GREEN); | support logic; impact depends on neighboring block and data f
| 2279 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2280 |         }); | support logic; impact depends on neighboring block and data flow. |
| 2281 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2282 | (blank) | blank separator; no runtime behavior. |
| 2283 |     fn render_ux_guide(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and co
| 2284 |         let (title, guide, legend) = match self.tab { | support logic; impact depends on neighboring block and
| 2285 |             Tab::Help => ( | support logic; impact depends on neighboring block and data flow. |
| 2286 |                 "Help", | support logic; impact depends on neighboring block and data flow. |
| 2287 |                 "Use esta aba para fluxo rapido de operacao, diagnostico e leitura da Leak Lab.", | support lo
| 2288 |                 "Comece por Help -> Cluster -> Engine -> Leaks e depois avance para CAN/UDS.", | protocol or I
| 2289 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2290 |             Tab::Cluster => ( | support logic; impact depends on neighboring block and data flow. |
| 2291 |                 "Cluster", | support logic; impact depends on neighboring block and data flow. |
| 2292 |                 "Use ignition -> throttle/brake -> direction to validate baseline vehicle response.", | suppor
| 2293 |                 "Lamps: RED=critical stop, AMBER=warning, ABS/ESP=assist active.", | support logic; impact dep
| 2294 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2295 |             Tab::Electrical => ( | support logic; impact depends on neighboring block and data flow. |
| 2296 |                 "Electrical", | support logic; impact depends on neighboring block and data flow. |
| 2297 |                 "Use esta aba para enxergar alimentacao, fusiveis, ramos ECU e referencias de chicote do model
| 2298 |                 "Valores de bitola/capacitancia sao baseline de simulacao e devem ser substituidos por dados O
| 2299 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2300 |             Tab::CanBus => ( | support logic; impact depends on neighboring block and data flow. |
| 2301 |                 "CAN Bus", | support logic; impact depends on neighboring block and data flow. |
| 2302 |                 "Start in Signals, then Trace, then Network for RCA. Freeze before exporting evidence.", | sup
| 2303 |                 "State colors: green=active, yellow=passive, red=bus-off/high error.", | support logic; impact
| 2304 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2305 |             Tab::Events => ( | support logic; impact depends on neighboring block and data flow. |
| 2306 |                 "Events", | support logic; impact depends on neighboring block and data flow. |
| 2307 |                 "Filter by severity and subsystem, then correlate timestamp with Faults/UDS actions.", | proto
| 2308 |                 "Icon color maps to severity; pause freezes feed for reading.", | support logic; impact depend
| 2309 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2310 |             Tab::EcuNet => ( | support logic; impact depends on neighboring block and data flow. |
| 2311 |                 "ECU Net", | support logic; impact depends on neighboring block and data flow. |
| 2312 |                 "Check node online state and boot stage before deep diagnostics.", | support logic; impact dep
| 2313 |                 "Degraded/offline nodes usually correlate with CAN or power faults.", | support logic; impact
| 2314 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2315 |             Tab::Engine => ( | support logic; impact depends on neighboring block and data flow. |
| 2316 |                 "Engine", | support logic; impact depends on neighboring block and data flow. |
| 2317 |                 "Validate thermal, pressure and torque limits under controlled load changes.", | support logic
```

```
| 2318 |                 "Watch red/yellow status first, then inspect numeric channels.", | support logic; impact depend
| 2319 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2320 |             Tab::Faults => ( | support logic; impact depends on neighboring block and data flow. |
| 2321 |                 "Faults", | support logic; impact depends on neighboring block and data flow. |
| 2322 |                 "Inject one fault at a time; observe Events/CAN/Engine and clear to compare recovery.", | suppo
| 2323 |                 "Use CLEAR after each test case to isolate cause/effect.", | support logic; impact depends on n
| 2324 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2325 |             Tab::Boot => ( | support logic; impact depends on neighboring block and data flow. |
| 2326 |                 "Boot", | support logic; impact depends on neighboring block and data flow. |
| 2327 |                 "Confirm startup sequence and timing before runtime validation.", | support logic; impact depen
| 2328 |                 "Blocked stage means readiness is not complete for higher-level tests.", | support logic; impac
| 2329 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2330 |             Tab::Implements => ( | support logic; impact depends on neighboring block and data flow. |
| 2331 |                 "Implements", | support logic; impact depends on neighboring block and data flow. |
| 2332 |                 "Operate PTO/hitch/aux valves incrementally and monitor hydraulic limits.", | support logic; in
| 2333 |                 "Large jumps in commands can mask root cause of instability.", | support logic; impact depends
| 2334 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2335 |             Tab::Params => ( | support logic; impact depends on neighboring block and data flow. |
| 2336 |                 "Params", | support logic; impact depends on neighboring block and data flow. |
| 2337 |                 "Edit one family of parameters at a time; sync back before changing scenario.", | support logic
| 2338 |                 "Use Sync from simulation to avoid stale values.", | support logic; impact depends on neighbori
| 2339 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2340 |             Tab::Sensors => ( | support logic; impact depends on neighboring block and data flow. |
| 2341 |                 "Sensors", | support logic; impact depends on neighboring block and data flow. |
| 2342 |                 "Validate GNSS/IMU/radar/lidar/camera consistency before AD validation.", | support logic; impa
| 2343 |                 "Prioritize confidence/quality metrics before control metrics.", | support logic; impact depend
| 2344 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2345 |             Tab::Autonomous => ( | support logic; impact depends on neighboring block and data flow. |
| 2346 |                 "AD", | support logic; impact depends on neighboring block and data flow. |
| 2347 |                 "Engage only after sensors and base dynamics are stable; tune ACC/LKA gradually.", | support lo
| 2348 |                 "TTC/THW and lane confidence are primary safety indicators.", | support logic; impact depends o
| 2349 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2350 |             Tab::V2X => ( | support logic; impact depends on neighboring block and data flow. |
| 2351 |                 "V2X", | support logic; impact depends on neighboring block and data flow. |
| 2352 |                 "Evaluate latency/loss first, then cooperative alerts impact on control.", | support logic; imp
| 2353 |                 "Higher packet loss can invalidate AD cooperative assumptions.", | support logic; impact depend
| 2354 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2355 |             Tab::Uds => ( | protocol or I/O critical; changes may impact external integration correctness. |
| 2356 |                 "UDS", | protocol or I/O critical; changes may impact external integration correctness. |
| 2357 |                 "Follow sequence: session -> security -> read/write/routine -> transfer exit.", | protocol or i
| 2358 |                 "Negative response 0x7F indicates rejected service/subfunction/conditions.", | support logic; i
| 2359 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2360 |             Tab::EcmLiveData => ( | support logic; impact depends on neighboring block and data flow. |
| 2361 |                 "ECM Live", | support logic; impact depends on neighboring block and data flow. |
| 2362 |                 "Detect -> Connect -> Retrieve -> Export. Keep mode and SA aligned.", | support logic; impact d
| 2363 |                 "In SIM mode, panels are mock; in LIVE mode, policy gates apply.", | support logic; impact depe
| 2364 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2365 |             Tab::LeakLab => ( | support logic; impact depends on neighboring block and data flow. |
| 2366 |                 "Leak Lab", | support logic; impact depends on neighboring block and data flow. |
| 2367 |                 "Select circuit, apply parameters, then run Scenario or Monte Carlo and review ranking.", | sup
| 2368 |                 "Use ASCII legend and temporal map to interpret pressure/risk/flow quickly.", | support logic;
| 2369 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2370 |             Tab::Plots => ( | support logic; impact depends on neighboring block and data flow. |
| 2371 |                 "Plots", | support logic; impact depends on neighboring block and data flow. |
| 2372 |                 "Use for before/after comparison around faults, AD engagement or UDS actions.", | protocol or i
| 2373 |                 "Trend shape matters more than single-point values.", | support logic; impact depends on neighb
| 2374 |             ), | support logic; impact depends on neighboring block and data flow. |
| 2375 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2376 | (blank) | blank separator; no runtime behavior. |
| 2377 |         egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data flow. |
| 2378 |             .fill(Color32::from_gray(16)) | support logic; impact depends on neighboring block and data flow.
| 2379 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2380 |                 ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data fl
| 2381 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2382 |                         RichText::new(format!("UX Guide - {}", title)) | support logic; impact depends on neigh
| 2383 |                             .size(11.6) | support logic; impact depends on neighboring block and data flow. |
| 2384 |                             .color(Color32::from_rgb(110, 180, 255)) | support logic; impact depends on neighbo
| 2385 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 2386 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 2387 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2388 |                     ui.checkbox(&mut self.ux_show_guide, "Show details"); | support logic; impact depends on ne
| 2389 |                     ui.checkbox(&mut self.ux_compact_mode, "Compact mode"); | support logic; impact depends on
| 2390 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2391 |                     self.render_ux_quick_actions(ui); | support logic; impact depends on neighboring block and
| 2392 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 2393 | (blank) | blank separator; no runtime behavior. |
```

```
| 2394 |                 for (color, msg) in self.ux_alerts() { | iteration logic; may alter timing, throughput, and sa
| 2395 |                     ui.add_space(4.0); | support logic; impact depends on neighboring block and data flow. |
| 2396 |                     ui.label(RichText::new(format!("• {}", msg)).size(10.3).color(color)); | support logic; imp
| 2397 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2398 | (blank) | blank separator; no runtime behavior. |
| 2399 |                 self.render_ux_context_strip(ui); | support logic; impact depends on neighboring block and data
| 2400 | (blank) | blank separator; no runtime behavior. |
| 2401 |                 if self.ux_show_guide { | runtime gate; condition changes can enable/disable critical paths. |
| 2402 |                     ui.add_space(4.0); | support logic; impact depends on neighboring block and data flow. |
| 2403 |                     ui.label(RichText::new(format!("How to use: {}", guide)).size(10.2)); | support logic; impa
| 2404 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2405 |                         RichText::new(format!("Legend/Interpretation: {}", legend)) | support logic; impact dep
| 2406 |                             .size(10.0) | support logic; impact depends on neighboring block and data flow. |
| 2407 |                             .color(Color32::from_gray(175)), | support logic; impact depends on neighboring bl
| 2408 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 2409 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2410 |             }); | support logic; impact depends on neighboring block and data flow. |
| 2411 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2412 | (blank) | blank separator; no runtime behavior. |
| 2413 |     fn render_ux_quick_actions(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavio
| 2414 |         match self.tab { | branch dispatcher; arm changes alter behavior class handling. |
| 2415 |             Tab::Help => { | support logic; impact depends on neighboring block and data flow. |
| 2416 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disa
| 2417 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data
| 2418 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2419 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2420 |             Tab::Cluster => { | support logic; impact depends on neighboring block and data flow. |
| 2421 |                 if ui.button("Key OFF").clicked() { | runtime gate; condition changes can enable/disable critic
| 2422 |                     self.cmds.push(Cmd::KeyOff); | support logic; impact depends on neighboring block and data
| 2423 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2424 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disa
| 2425 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data
| 2426 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2427 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2428 |             Tab::Electrical => { | support logic; impact depends on neighboring block and data flow. |
| 2429 |                 if ui.button("Toggle HVAC").clicked() { | runtime gate; condition changes can enable/disable c
| 2430 |                     self.cmds.push(Cmd::ToggleHvac); | support logic; impact depends on neighboring block and
| 2431 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2432 |                 if ui.button("Reset fuses").clicked() { | runtime gate; condition changes can enable/disable c
| 2433 |                     self.cmds.push(Cmd::ResetBcmFuses); | support logic; impact depends on neighboring block a
| 2434 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2435 |                 if ui.button("Toggle work lights").clicked() { | runtime gate; condition changes can enable/di
| 2436 |                     self.cmds.push(Cmd::ToggleWorkLights); | support logic; impact depends on neighboring bloc
| 2437 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2438 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2439 |             Tab::Events => { | support logic; impact depends on neighboring block and data flow. |
| 2440 |                 if ui.button("Clear events").clicked() { | runtime gate; condition changes can enable/disable
| 2441 |                     self.events.clear(); | support logic; impact depends on neighboring block and data flow. |
| 2442 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2443 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2444 |             Tab::CanBus => { | support logic; impact depends on neighboring block and data flow. |
| 2445 |                 if ui.button("Clear CAN views").clicked() { | runtime gate; condition changes can enable/disab
| 2446 |                     self.sig_map.clear(); | support logic; impact depends on neighboring block and data flow.
| 2447 |                     self.trace_snap.clear(); | support logic; impact depends on neighboring block and data flo
| 2448 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2449 |                 if ui.button("Clear CAN injections").clicked() { | runtime gate; condition changes can enable/
| 2450 |                     self.cmds.push(Cmd::CanClearAllInjections); | support logic; impact depends on neighboring
| 2451 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2452 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2453 |             Tab::EcuNet => { | support logic; impact depends on neighboring block and data flow. |
| 2454 |                 if ui.button("Key OFF").clicked() { | runtime gate; condition changes can enable/disable critic
| 2455 |                     self.cmds.push(Cmd::KeyOff); | support logic; impact depends on neighboring block and data
| 2456 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2457 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disa
| 2458 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data
| 2459 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2460 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2461 |             Tab::Engine => { | support logic; impact depends on neighboring block and data flow. |
| 2462 |                 if ui.button("Clear DTCs").clicked() { | runtime gate; condition changes can enable/disable cri
| 2463 |                     self.cmds.push(Cmd::ClearDtcs); | support logic; impact depends on neighboring block and da
| 2464 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2465 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disa
| 2466 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data
| 2467 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2468 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2469 |             Tab::Faults => { | support logic; impact depends on neighboring block and data flow. |
```

```
| 2470 |                 if ui.button("Inject selected").clicked() { | runtime gate; condition changes can enable/disabl
| 2471 |                     self.cmds.push(Cmd::InjectFault); | support logic; impact depends on neighboring block and
| 2472 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2473 |                 if ui.button("Clear faults").clicked() { | runtime gate; condition changes can enable/disable c
| 2474 |                     self.cmds.push(Cmd::ClearFaults); | support logic; impact depends on neighboring block and
| 2475 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2476 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2477 |             Tab::Boot => { | support logic; impact depends on neighboring block and data flow. |
| 2478 |                 if ui.button("Key advance").clicked() { | runtime gate; condition changes can enable/disable cr
| 2479 |                     self.cmds.push(Cmd::KeyAdvance); | support logic; impact depends on neighboring block and c
| 2480 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2481 |                 if ui.button("Key OFF").clicked() { | runtime gate; condition changes can enable/disable critic
| 2482 |                     self.cmds.push(Cmd::KeyOff); | support logic; impact depends on neighboring block and data
| 2483 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2484 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2485 |             Tab::Implements => { | support logic; impact depends on neighboring block and data flow. |
| 2486 |                 if ui.button("PTO OFF").clicked() { | runtime gate; condition changes can enable/disable critic
| 2487 |                     self.cmds.push(Cmd::SetPtoMode(PtoMode::Off)); | support logic; impact depends on neighbori
| 2488 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2489 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disal
| 2490 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data f
| 2491 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2492 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2493 |             Tab::Params => { | support logic; impact depends on neighboring block and data flow. |
| 2494 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disal
| 2495 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data f
| 2496 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2497 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2498 |             Tab::Sensors => { | support logic; impact depends on neighboring block and data flow. |
| 2499 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disal
| 2500 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data f
| 2501 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2502 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2503 |             Tab::Autonomous => { | support logic; impact depends on neighboring block and data flow. |
| 2504 |                 if ui | runtime gate; condition changes can enable/disable critical paths. |
| 2505 |                     .button(if self.bench.ad.engaged { "Disengage AD" } else { "Engage AD" }) | support logic;
| 2506 |                     .clicked() | support logic; impact depends on neighboring block and data flow. |
| 2507 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2508 |                     self.cmds.push(if self.bench.ad.engaged { | support logic; impact depends on neighboring bl
| 2509 |                         Cmd::DisengageAD | support logic; impact depends on neighboring block and data flow. |
| 2510 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2511 |                         Cmd::EngageAD | support logic; impact depends on neighboring block and data flow. |
| 2512 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2513 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2514 |                 if ui.button("Clear path").clicked() { | runtime gate; condition changes can enable/disable cri
| 2515 |                     self.cmds.push(Cmd::ClearWaypoints); | support logic; impact depends on neighboring block a
| 2516 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2517 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2518 |             Tab::V2X => { | support logic; impact depends on neighboring block and data flow. |
| 2519 |                 if ui | runtime gate; condition changes can enable/disable critical paths. |
| 2520 |                     .add_enabled(self.bench.telematics.ota_available, Button::new("Start OTA")) | support logic
| 2521 |                     .clicked() | support logic; impact depends on neighboring block and data flow. |
| 2522 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2523 |                     self.cmds.push(Cmd::StartOta); | support logic; impact depends on neighboring block and dat
| 2524 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2525 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disal
| 2526 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data f
| 2527 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2528 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2529 |             Tab::Uds => { | protocol or I/O critical; changes may impact external integration correctness. |
| 2530 |                 if ui.button("Clear UDS log").clicked() { | runtime gate; condition changes can enable/disable
| 2531 |                     self.uds_log.clear(); | protocol or I/O critical; changes may impact external integration c
| 2532 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2533 |                 if ui.button("Default Session").clicked() { | runtime gate; condition changes can enable/disabl
| 2534 |                     self.uds_input = "10 01".into(); | protocol or I/O critical; changes may impact external in
| 2535 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2536 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2537 |             Tab::EcmLiveData => { | support logic; impact depends on neighboring block and data flow. |
| 2538 |                 if ui.button("Clear live snapshot").clicked() { | runtime gate; condition changes can enable/di
| 2539 |                     self.ecm_live_snapshot = io::ecm_params::EcmSnapshot::default(); | support logic; impact de
| 2540 |                     self.ecm_live_frames_total = 0; | support logic; impact depends on neighboring block and da
| 2541 |                     self.ecm_live_last_update_ms = 0; | support logic; impact depends on neighboring block and
| 2542 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2543 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2544 |             Tab::LeakLab => { | support logic; impact depends on neighboring block and data flow. |
| 2545 |                 if ui.button("Clear leak outputs").clicked() { | runtime gate; condition changes can enable/dis
```

```
| 2546 |                     self.cmds.push(Cmd::LeakClearScenarioOutputs); | support logic; impact depends on neighbori
| 2547 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2548 |                 if ui.button("Reset leak sim").clicked() { | runtime gate; condition changes can enable/disable
| 2549 |                     self.cmds.push(Cmd::LeakResetSimulation); | support logic; impact depends on neighboring b
| 2550 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2551 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2552 |             Tab::Plots => { | support logic; impact depends on neighboring block and data flow. |
| 2553 |                 if ui.button("Reset simulation").clicked() { | runtime gate; condition changes can enable/disal
| 2554 |                     self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data f
| 2555 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2556 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2557 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2558 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2559 | (blank) | blank separator; no runtime behavior. |
| 2560 |     fn electrical_wire_mm2(current_a: f64) -> f64 { | function signature/body; changes affect API behavior and
| 2561 |         if current_a <= 5.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 2562 |             0.75 | support logic; impact depends on neighboring block and data flow. |
| 2563 |         } else if current_a <= 10.0 { | support logic; impact depends on neighboring block and data flow. |
| 2564 |             1.0 | support logic; impact depends on neighboring block and data flow. |
| 2565 |         } else if current_a <= 15.0 { | support logic; impact depends on neighboring block and data flow. |
| 2566 |             1.5 | support logic; impact depends on neighboring block and data flow. |
| 2567 |         } else if current_a <= 25.0 { | support logic; impact depends on neighboring block and data flow. |
| 2568 |             2.5 | support logic; impact depends on neighboring block and data flow. |
| 2569 |         } else if current_a <= 40.0 { | support logic; impact depends on neighboring block and data flow. |
| 2570 |             4.0 | support logic; impact depends on neighboring block and data flow. |
| 2571 |         } else if current_a <= 60.0 { | support logic; impact depends on neighboring block and data flow. |
| 2572 |             6.0 | support logic; impact depends on neighboring block and data flow. |
| 2573 |         } else if current_a <= 100.0 { | support logic; impact depends on neighboring block and data flow. |
| 2574 |             16.0 | support logic; impact depends on neighboring block and data flow. |
| 2575 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 2576 |             25.0 | support logic; impact depends on neighboring block and data flow. |
| 2577 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2578 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2579 | (blank) | blank separator; no runtime behavior. |
| 2580 |     fn electrical_cap_nf_per_m(gauge_mm2: f64, twisted_pair: bool) -> f64 { | function signature/body; changes
| 2581 |         if twisted_pair { | runtime gate; condition changes can enable/disable critical paths. |
| 2582 |             0.05 | support logic; impact depends on neighboring block and data flow. |
| 2583 |         } else if gauge_mm2 <= 1.0 { | support logic; impact depends on neighboring block and data flow. |
| 2584 |             0.09 | support logic; impact depends on neighboring block and data flow. |
| 2585 |         } else if gauge_mm2 <= 2.5 { | support logic; impact depends on neighboring block and data flow. |
| 2586 |             0.11 | support logic; impact depends on neighboring block and data flow. |
| 2587 |         } else if gauge_mm2 <= 6.0 { | support logic; impact depends on neighboring block and data flow. |
| 2588 |             0.13 | support logic; impact depends on neighboring block and data flow. |
| 2589 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 2590 |             0.16 | support logic; impact depends on neighboring block and data flow. |
| 2591 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2592 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2593 | (blank) | blank separator; no runtime behavior. |
| 2594 |     fn electrical_node_online(&self, sa: u8) -> bool { | function signature/body; changes affect API behavior a
| 2595 |         self.bench | support logic; impact depends on neighboring block and data flow. |
| 2596 |             .boot | support logic; impact depends on neighboring block and data flow. |
| 2597 |             .ecu_by_sa(sa) | support logic; impact depends on neighboring block and data flow. |
| 2598 |             .is_some_and(\|ecu\| ecu.is_online()) | support logic; impact depends on neighboring block and data
| 2599 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2600 | (blank) | blank separator; no runtime behavior. |
| 2601 |     fn tab_ecm_live_data(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and c
| 2602 |         ui.heading("ECM-Live Data"); | support logic; impact depends on neighboring block and data flow. |
| 2603 |         ui.add_space(6.0); | support logic; impact depends on neighboring block and data flow. |
| 2604 |         let live_mode = matches!(self.hw_cfg.mode, io::hw::HwMode::Live); | support logic; impact depends on ne
| 2605 | (blank) | blank separator; no runtime behavior. |
| 2606 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 2607 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 2608 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2609 | (blank) | blank separator; no runtime behavior. |
| 2610 |                 if !live_mode { | runtime gate; condition changes can enable/disable critical paths. |
| 2611 |                     ui.colored_label( | support logic; impact depends on neighboring block and data flow. |
| 2612 |                         Color32::YELLOW, | support logic; impact depends on neighboring block and data flow. |
| 2613 |                         "SIM mode active: showing mock ECM view. Switch to --hw-mode=live for hardware Detect/C
| 2614 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 2615 |                     if self.ecm_detected_sas.is_empty() { | runtime gate; condition changes can enable/disable
| 2616 |                         self.ecm_detected_sas = vec![0x00]; | support logic; impact depends on neighboring bloc
| 2617 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2618 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2619 | (blank) | blank separator; no runtime behavior. |
| 2620 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2621 |             if ui.add_enabled(live_mode, Button::new("Detect")).clicked() { | runtime gate; condition changes c
```

```
| 2622 |                 match io::live_runner::detect_ecms(&self.hw_cfg, Duration::from_secs(3)) { | branch dispatcher
| 2623 |                     Ok(res) => { | support logic; impact depends on neighboring block and data flow. |
| 2624 |                         self.ecm_detected_sas = res.source_addresses; | support logic; impact depends on neighl
| 2625 |                         self.ecm_selected_idx = 0; | support logic; impact depends on neighboring block and dat
| 2626 |                         self.ecm_connected = false; | support logic; impact depends on neighboring block and da
| 2627 |                         if self.ecm_detected_sas.is_empty() { | runtime gate; condition changes can enable/disa
| 2628 |                             self.ecm_live_status = | support logic; impact depends on neighboring block and dat
| 2629 |                                 "Detect completed: no ECM found in timeout window.".into(); | support logic; in
| 2630 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 2631 |                             self.ecm_live_status = format!( | support logic; impact depends on neighboring bloc
| 2632 |                                 "Detect completed: {} ECM source address(es) found.", | support logic; impact c
| 2633 |                                 self.ecm_detected_sas.len() | support logic; impact depends on neighboring bloc
| 2634 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 2635 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 2636 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2637 |                     Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 2638 |                         self.ecm_live_status = format!("Detect failed: {e}"); | support logic; impact depends c
| 2639 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2640 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2641 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2642 | (blank) | blank separator; no runtime behavior. |
| 2643 |             let can_connect = live_mode && !self.ecm_detected_sas.is_empty(); | support logic; impact depends c
| 2644 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 2645 |                 .add_enabled(can_connect, Button::new("Connect")) | support logic; impact depends on neighborin
| 2646 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 2647 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2648 |                 let selected_sa = self.ecm_detected_sas.get(self.ecm_selected_idx).copied(); | support logic; i
| 2649 |                 let target_sa = if self.hw_cfg.can_interface.is_some() { | support logic; impact depends on nei
| 2650 |                     selected_sa | support logic; impact depends on neighboring block and data flow. |
| 2651 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 2652 |                     None | support logic; impact depends on neighboring block and data flow. |
| 2653 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2654 | (blank) | blank separator; no runtime behavior. |
| 2655 |                 match io::live_runner::connect_ecm(&self.hw_cfg, target_sa) { | branch dispatcher; arm changes
| 2656 |                     Ok(()) => { | support logic; impact depends on neighboring block and data flow. |
| 2657 |                         self.ecm_connected = true; | support logic; impact depends on neighboring block and dat
| 2658 |                         self.ecm_live_status = "Connect successful. Ready to Retrieve Data.".into(); | support
| 2659 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2660 |                     Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 2661 |                         self.ecm_connected = false; | support logic; impact depends on neighboring block and da
| 2662 |                         self.ecm_live_status = format!("Connect failed: {e}"); | support logic; impact depends
| 2663 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2664 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2665 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2666 | (blank) | blank separator; no runtime behavior. |
| 2667 |             let can_retrieve = live_mode && self.ecm_connected && self.ecm_live_feed.is_none(); | support logic
| 2668 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 2669 |                 .add_enabled(can_retrieve, Button::new("Retrieve Data")) | support logic; impact depends on nei
| 2670 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 2671 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2672 |                 let selected_sa = self.ecm_detected_sas.get(self.ecm_selected_idx).copied(); | support logic; i
| 2673 |                 let target_sa = if self.hw_cfg.can_interface.is_some() { | support logic; impact depends on nei
| 2674 |                     selected_sa | support logic; impact depends on neighboring block and data flow. |
| 2675 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 2676 |                     None | support logic; impact depends on neighboring block and data flow. |
| 2677 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2678 | (blank) | blank separator; no runtime behavior. |
| 2679 |                 match io::live_runner::start_retrieve_data(self.hw_cfg.clone(), target_sa) { | branch dispatche
| 2680 |                     Ok(feed) => { | support logic; impact depends on neighboring block and data flow. |
| 2681 |                         self.ecm_live_feed = Some(feed); | support logic; impact depends on neighboring block a
| 2682 |                         self.ecm_live_status = | support logic; impact depends on neighboring block and data f
| 2683 |                             "Retrieve started. Real-time parameter updates are active.".into(); | support logic
| 2684 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2685 |                     Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 2686 |                         self.ecm_live_status = format!("Retrieve failed: {e}"); | support logic; impact depends
| 2687 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2688 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2689 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2690 | (blank) | blank separator; no runtime behavior. |
| 2691 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 2692 |                 .add_enabled(self.ecm_live_feed.is_some(), Button::new("Stop")) | support logic; impact depends
| 2693 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 2694 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2695 |                 if let Some(feed) = &self.ecm_live_feed { | runtime gate; condition changes can enable/disable
| 2696 |                     feed.stop(); | support logic; impact depends on neighboring block and data flow. |
| 2697 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 2698 |                 self.ecm_live_feed = None; | support logic; impact depends on neighboring block and data flow.
| 2699 |                 self.ecm_live_status = "Retrieve stopped by operator.".into(); | support logic; impact depends
| 2700 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2701 | (blank) | blank separator; no runtime behavior. |
| 2702 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 2703 |                 .add_enabled(self.ecm_live_feed.is_some(), Button::new("Export CSV")) | support logic; impact d
| 2704 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 2705 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2706 |                 if let Some(feed) = &self.ecm_live_feed { | runtime gate; condition changes can enable/disable
| 2707 |                     let ts = chrono::Local::now().format("%Y%m%d_%H%M%S").to_string(); | support logic; impact
| 2708 |                     let name = format!("ecm_live_{}.csv", ts); | support logic; impact depends on neighboring b
| 2709 |                     if let Some(path) = Self::pick_save_path(&name, "csv") { | runtime gate; condition changes
| 2710 |                         match feed.export_csv(&path) { | branch dispatcher; arm changes alter behavior class ha
| 2711 |                             Ok(()) => { | support logic; impact depends on neighboring block and data flow. |
| 2712 |                                 self.ecm_live_status = format!("CSV exported to {}", path); | support logic; im
| 2713 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 2714 |                             Err(e) => { | support logic; impact depends on neighboring block and data flow. |
| 2715 |                                 self.ecm_live_status = format!("CSV export failed: {e}"); | support logic; impa
| 2716 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 2717 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 2718 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2719 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2720 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2721 |         }); | support logic; impact depends on neighboring block and data flow. |
| 2722 | (blank) | blank separator; no runtime behavior. |
| 2723 |         ui.add_space(8.0); | support logic; impact depends on neighboring block and data flow. |
| 2724 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2725 |             ui.label("Detected ECM SA:"); | support logic; impact depends on neighboring block and data flow.
| 2726 |             if self.ecm_detected_sas.is_empty() { | runtime gate; condition changes can enable/disable critical
| 2727 |                 ui.label("(none)"); | support logic; impact depends on neighboring block and data flow. |
| 2728 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 2729 |                 let mut selected = self.ecm_selected_idx.min(self.ecm_detected_sas.len() - 1); | support logic
| 2730 |                 ComboBox::from_id_salt("ecm_live::sa_combo") | support logic; impact depends on neighboring bl
| 2731 |                     .selected_text(format!("0x{:02X}", self.ecm_detected_sas[selected])) | support logic; impac
| 2732 |                     .show_ui(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 2733 |                         for (i, sa) in self.ecm_detected_sas.iter().enumerate() { | iteration logic; may alter
| 2734 |                             ui.selectable_value(&mut selected, i, format!("0x{:02X}", sa)); | support logic; im
| 2735 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 2736 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2737 |                 self.ecm_selected_idx = selected; | support logic; impact depends on neighboring block and data
| 2738 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2739 |         }); | support logic; impact depends on neighboring block and data flow. |
| 2740 | (blank) | blank separator; no runtime behavior. |
| 2741 |         ui.add_space(8.0); | support logic; impact depends on neighboring block and data flow. |
| 2742 |         ui.label(format!("Status: {}", self.ecm_live_status)); | support logic; impact depends on neighboring b
| 2743 |         ui.label(format!( | support logic; impact depends on neighboring block and data flow. |
| 2744 |             "Connected: {} \| Streaming: {} \| Frames: {} \| Last update(ms): {}", | support logic; impact depe
| 2745 |             self.ecm_connected, | support logic; impact depends on neighboring block and data flow. |
| 2746 |             self.ecm_live_feed.is_some(), | support logic; impact depends on neighboring block and data flow.
| 2747 |             self.ecm_live_frames_total, | support logic; impact depends on neighboring block and data flow. |
| 2748 |             self.ecm_live_last_update_ms | support logic; impact depends on neighboring block and data flow. |
| 2749 |         )); | support logic; impact depends on neighboring block and data flow. |
| 2750 | (blank) | blank separator; no runtime behavior. |
| 2751 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2752 |         if let Some(feed) = &self.ecm_live_feed { | runtime gate; condition changes can enable/disable critical
| 2753 |             let points = feed.recent_points(300); | support logic; impact depends on neighboring block and data
| 2754 |             if !points.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 2755 |                 let mut rpm_min = f64::INFINITY; | support logic; impact depends on neighboring block and data
| 2756 |                 let mut rpm_max = f64::NEG_INFINITY; | support logic; impact depends on neighboring block and d
| 2757 |                 let mut rpm_sum = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 2758 |                 let mut rpm_count = 0usize; | support logic; impact depends on neighboring block and data flow
| 2759 |                 let mut cool_max = f64::NEG_INFINITY; | support logic; impact depends on neighboring block and
| 2760 |                 let mut oil_min = f64::INFINITY; | support logic; impact depends on neighboring block and data
| 2761 | (blank) | blank separator; no runtime behavior. |
| 2762 |                 for p in &points { | iteration logic; may alter timing, throughput, and safety constraints. |
| 2763 |                     if let Some(r) = p.engine_speed_rpm { | runtime gate; condition changes can enable/disable
| 2764 |                         rpm_min = rpm_min.min(r); | support logic; impact depends on neighboring block and data
| 2765 |                         rpm_max = rpm_max.max(r); | support logic; impact depends on neighboring block and data
| 2766 |                         rpm_sum += r; | support logic; impact depends on neighboring block and data flow. |
| 2767 |                         rpm_count += 1; | support logic; impact depends on neighboring block and data flow. |
| 2768 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2769 |                     if let Some(c) = p.coolant_temp_c { | runtime gate; condition changes can enable/disable c
| 2770 |                         cool_max = cool_max.max(c); | support logic; impact depends on neighboring block and d
| 2771 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2772 |                     if let Some(o) = p.oil_pressure_kpa { | runtime gate; condition changes can enable/disable
| 2773 |                         oil_min = oil_min.min(o); | support logic; impact depends on neighboring block and data
```

```
| 2774 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2775 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2776 | (blank) | blank separator; no runtime behavior. |
| 2777 |                 ui.label("Post-analysis dashboard (rolling window)"); | support logic; impact depends on neighb
| 2778 |                 Grid::new("ecm_live::dashboard_grid").show(ui, \|ui\| { | support logic; impact depends on neig
| 2779 |                     ui.label("Samples"); | support logic; impact depends on neighboring block and data flow. |
| 2780 |                     ui.label(format!("{}", points.len())); | support logic; impact depends on neighboring bloc
| 2781 |                     ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2782 | (blank) | blank separator; no runtime behavior. |
| 2783 |                     ui.label("RPM Min"); | support logic; impact depends on neighboring block and data flow. |
| 2784 |                     ui.label(if rpm_count > 0 { | support logic; impact depends on neighboring block and data
| 2785 |                         format!("{:.1}", rpm_min) | support logic; impact depends on neighboring block and data
| 2786 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2787 |                         "-".to_string() | support logic; impact depends on neighboring block and data flow. |
| 2788 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2789 |                     ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2790 | (blank) | blank separator; no runtime behavior. |
| 2791 |                     ui.label("RPM Avg"); | support logic; impact depends on neighboring block and data flow. |
| 2792 |                     ui.label(if rpm_count > 0 { | support logic; impact depends on neighboring block and data
| 2793 |                         format!("{:.1}", rpm_sum / rpm_count as f64) | support logic; impact depends on neighb
| 2794 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2795 |                         "-".to_string() | support logic; impact depends on neighboring block and data flow. |
| 2796 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2797 |                     ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2798 | (blank) | blank separator; no runtime behavior. |
| 2799 |                     ui.label("RPM Max"); | support logic; impact depends on neighboring block and data flow. |
| 2800 |                     ui.label(if rpm_count > 0 { | support logic; impact depends on neighboring block and data
| 2801 |                         format!("{:.1}", rpm_max) | support logic; impact depends on neighboring block and data
| 2802 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2803 |                         "-".to_string() | support logic; impact depends on neighboring block and data flow. |
| 2804 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2805 |                     ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2806 | (blank) | blank separator; no runtime behavior. |
| 2807 |                     ui.label("Coolant Peak C"); | support logic; impact depends on neighboring block and data
| 2808 |                     ui.label(if cool_max.is_finite() { | support logic; impact depends on neighboring block an
| 2809 |                         format!("{:.1}", cool_max) | support logic; impact depends on neighboring block and da
| 2810 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2811 |                         "-".to_string() | support logic; impact depends on neighboring block and data flow. |
| 2812 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2813 |                     ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2814 | (blank) | blank separator; no runtime behavior. |
| 2815 |                     ui.label("Oil Pressure Min kPa"); | support logic; impact depends on neighboring block and
| 2816 |                     ui.label(if oil_min.is_finite() { | support logic; impact depends on neighboring block and
| 2817 |                         format!("{:.1}", oil_min) | support logic; impact depends on neighboring block and data
| 2818 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2819 |                         "-".to_string() | support logic; impact depends on neighboring block and data flow. |
| 2820 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 2821 |                     ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2822 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 2823 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2824 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2825 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2826 | (blank) | blank separator; no runtime behavior. |
| 2827 |         ui.label("Real-time ECM Snapshot:"); | support logic; impact depends on neighboring block and data flo
| 2828 |         Grid::new("ecm_live::snapshot_grid").show(ui, \|ui\| { | support logic; impact depends on neighboring
| 2829 |             ui.label("Engine Speed"); | support logic; impact depends on neighboring block and data flow. |
| 2830 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2831 |                 self.ecm_live_snapshot | support logic; impact depends on neighboring block and data flow. |
| 2832 |                     .engine_speed_rpm | support logic; impact depends on neighboring block and data flow. |
| 2833 |                     .map_or("-".into(), \|v\| format!("{v:.1} rpm")), | support logic; impact depends on neigh
| 2834 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2835 |             ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2836 | (blank) | blank separator; no runtime behavior. |
| 2837 |             ui.label("Accel Pedal"); | support logic; impact depends on neighboring block and data flow. |
| 2838 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2839 |                 self.ecm_live_snapshot | support logic; impact depends on neighboring block and data flow. |
| 2840 |                     .accel_pedal_pct | support logic; impact depends on neighboring block and data flow. |
| 2841 |                     .map_or("-".into(), \|v\| format!("{v:.1} %")), | support logic; impact depends on neighbo
| 2842 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2843 |             ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2844 | (blank) | blank separator; no runtime behavior. |
| 2845 |             ui.label("Coolant Temp"); | support logic; impact depends on neighboring block and data flow. |
| 2846 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2847 |                 self.ecm_live_snapshot | support logic; impact depends on neighboring block and data flow. |
| 2848 |                     .coolant_temp_c | support logic; impact depends on neighboring block and data flow. |
| 2849 |                     .map_or("-".into(), \|v\| format!("{v:.1} C")), | support logic; impact depends on neighbo
```

```
| 2850 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2851 |             ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2852 | (blank) | blank separator; no runtime behavior. |
| 2853 |             ui.label("Fuel Temp"); | support logic; impact depends on neighboring block and data flow. |
| 2854 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2855 |                 self.ecm_live_snapshot | support logic; impact depends on neighboring block and data flow. |
| 2856 |                     .fuel_temp_c | support logic; impact depends on neighboring block and data flow. |
| 2857 |                     .map_or("-".into(), \|v\| format!("{v:.1} C")), | support logic; impact depends on neighbo
| 2858 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2859 |             ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2860 | (blank) | blank separator; no runtime behavior. |
| 2861 |             ui.label("Oil Pressure"); | support logic; impact depends on neighboring block and data flow. |
| 2862 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2863 |                 self.ecm_live_snapshot | support logic; impact depends on neighboring block and data flow. |
| 2864 |                     .oil_pressure_kpa | support logic; impact depends on neighboring block and data flow. |
| 2865 |                     .map_or("-".into(), \|v\| format!("{v:.1} kPa")), | support logic; impact depends on neighb
| 2866 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2867 |             ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2868 | (blank) | blank separator; no runtime behavior. |
| 2869 |             ui.label("Last PGN"); | support logic; impact depends on neighboring block and data flow. |
| 2870 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2871 |                 self.ecm_live_snapshot | support logic; impact depends on neighboring block and data flow. |
| 2872 |                     .last_seen_pgn | support logic; impact depends on neighboring block and data flow. |
| 2873 |                     .map_or("-".into(), \|v\| format!("{v}")), | support logic; impact depends on neighboring b
| 2874 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2875 |             ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2876 | (blank) | blank separator; no runtime behavior. |
| 2877 |             ui.label("Source Address"); | support logic; impact depends on neighboring block and data flow. |
| 2878 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2879 |                 self.ecm_live_snapshot | support logic; impact depends on neighboring block and data flow. |
| 2880 |                     .source_address | support logic; impact depends on neighboring block and data flow. |
| 2881 |                     .map_or("-".into(), \|v\| format!("0x{v:02X}")), | support logic; impact depends on neighbo
| 2882 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2883 |             ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 2884 |         }); | support logic; impact depends on neighboring block and data flow. |
| 2885 | (blank) | blank separator; no runtime behavior. |
| 2886 |                 self.ecm_treeview(ui, !live_mode); | support logic; impact depends on neighboring block and da
| 2887 |             }); | support logic; impact depends on neighboring block and data flow. |
| 2888 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2889 | (blank) | blank separator; no runtime behavior. |
| 2890 |     fn toolbar(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flo
| 2891 |         ui.push_id("ui::toolbar", \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 2892 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 2893 |             ui.add_space(6.0); | support logic; impact depends on neighboring block and data flow. |
| 2894 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2895 |                 RichText::new("■ ECU BENCH v3.0") | support logic; impact depends on neighboring block and dat
| 2896 |                     .size(17.0) | support logic; impact depends on neighboring block and data flow. |
| 2897 |                     .color(Color32::from_rgb(80, 155, 255)) | support logic; impact depends on neighboring bloc
| 2898 |                     .strong(), | support logic; impact depends on neighboring block and data flow. |
| 2899 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2900 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2901 |             // IGN key | comment/documentation; changing affects understanding and intent traceability. |
| 2902 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2903 |                 RichText::new("IGN") | support logic; impact depends on neighboring block and data flow. |
| 2904 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 2905 |                     .color(Color32::from_gray(150)), | support logic; impact depends on neighboring block and d
| 2906 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2907 |             let ign = self.bench.ignition(); | support logic; impact depends on neighboring block and data flow
| 2908 |             for (lbl, target) in [ | iteration logic; may alter timing, throughput, and safety constraints. |
| 2909 |                 ("OFF", IgnitionState::Off), | support logic; impact depends on neighboring block and data flow
| 2910 |                 ("ACC", IgnitionState::Accessory), | support logic; impact depends on neighboring block and dat
| 2911 |                 ("ON", IgnitionState::On), | support logic; impact depends on neighboring block and data flow.
| 2912 |                 ("START", IgnitionState::Cranking), | support logic; impact depends on neighboring block and da
| 2913 |             ] { | support logic; impact depends on neighboring block and data flow. |
| 2914 |                 let cur = ign == target; | support logic; impact depends on neighboring block and data flow. |
| 2915 |                 let col = if cur { | support logic; impact depends on neighboring block and data flow. |
| 2916 |                     match target { | branch dispatcher; arm changes alter behavior class handling. |
| 2917 |                         IgnitionState::Off => Color32::from_gray(110), | support logic; impact depends on neigh
| 2918 |                         IgnitionState::Accessory => Color32::YELLOW, | support logic; impact depends on neighbo
| 2919 |                         IgnitionState::On => Color32::LIGHT_BLUE, | support logic; impact depends on neighborin
| 2920 |                         IgnitionState::Cranking => Color32::from_rgb(255, 130, 40), | support logic; impact dep
| 2921 |                         IgnitionState::Running => Color32::GREEN, | support logic; impact depends on neighborin
| 2922 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2923 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 2924 |                     Color32::from_gray(40) | support logic; impact depends on neighboring block and data flow.
| 2925 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 2926 |                 let btn = Button::new(RichText::new(lbl).size(11.0).color(if cur { | support logic; impact depe
| 2927 |                     Color32::BLACK | support logic; impact depends on neighboring block and data flow. |
| 2928 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 2929 |                     Color32::from_gray(155) | support logic; impact depends on neighboring block and data flow
| 2930 |                 })) | support logic; impact depends on neighboring block and data flow. |
| 2931 |                 .fill(col) | support logic; impact depends on neighboring block and data flow. |
| 2932 |                 .min_size(Vec2::new(48.0, 26.0)); | support logic; impact depends on neighboring block and data
| 2933 |                 if ui.add(btn).clicked() { | runtime gate; condition changes can enable/disable critical paths
| 2934 |                     match target { | branch dispatcher; arm changes alter behavior class handling. |
| 2935 |                         IgnitionState::Off => self.cmds.push(Cmd::KeyOff), | support logic; impact depends on
| 2936 |                         _ => { | support logic; impact depends on neighboring block and data flow. |
| 2937 |                             for _ in 0..5 { | iteration logic; may alter timing, throughput, and safety constra
| 2938 |                                 if self.bench.ignition() == target { | runtime gate; condition changes can enal
| 2939 |                                     break; | support logic; impact depends on neighboring block and data flow.
| 2940 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 2941 |                                 self.cmds.push(Cmd::KeyAdvance); | support logic; impact depends on neighborin
| 2942 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 2943 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 2944 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2945 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2946 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2947 |             let run = self.bench.engine_running(); | support logic; impact depends on neighboring block and da
| 2948 |             let ad_locked = self.bench.ad.engaged; | support logic; impact depends on neighboring block and da
| 2949 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2950 |                 RichText::new(if run { "● RUN" } else { "■ OFF" }) | support logic; impact depends on neighbor
| 2951 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 2952 |                     .color(if run { | support logic; impact depends on neighboring block and data flow. |
| 2953 |                         Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 2954 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 2955 |                         Color32::from_gray(80) | support logic; impact depends on neighboring block and data f
| 2956 |                     }), | support logic; impact depends on neighboring block and data flow. |
| 2957 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2958 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2959 |             // Throttle | comment/documentation; changing affects understanding and intent traceability. |
| 2960 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2961 |                 RichText::new("THROT") | support logic; impact depends on neighboring block and data flow. |
| 2962 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 2963 |                     .color(Color32::from_gray(145)), | support logic; impact depends on neighboring block and
| 2964 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2965 |             let mut thr = self.throttle_target; | support logic; impact depends on neighboring block and data
| 2966 |             let thr_resp = ui.add_enabled( | support logic; impact depends on neighboring block and data flow.
| 2967 |                 !ad_locked, | support logic; impact depends on neighboring block and data flow. |
| 2968 |                 Slider::new(&mut thr, 0.0..=1.0) | support logic; impact depends on neighboring block and data
| 2969 |                     .show_value(false) | support logic; impact depends on neighboring block and data flow. |
| 2970 |                     .trailing_fill(true), | support logic; impact depends on neighboring block and data flow.
| 2971 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2972 |             if ad_locked { | runtime gate; condition changes can enable/disable critical paths. |
| 2973 |                 let _ = thr_resp.on_disabled_hover_text("Throttle manual bloqueado enquanto o AD estiver ativo
| 2974 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 2975 |                 let thr_changed = (thr - self.throttle_cmd_last).abs() > 0.01; | support logic; impact depends
| 2976 |                 let thr_due = self.ticks.saturating_sub(self.throttle_cmd_last_tick) >= 2; | support logic; imp
| 2977 |                 if thr != self.throttle_target && (thr_changed \|\| thr_due) { | runtime gate; condition change
| 2978 |                     self.cmds.push(Cmd::SetThrottle(thr)); | support logic; impact depends on neighboring bloc
| 2979 |                     self.throttle_cmd_last = thr; | support logic; impact depends on neighboring block and data
| 2980 |                     self.throttle_cmd_last_tick = self.ticks; | support logic; impact depends on neighboring b
| 2981 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2982 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 2983 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2984 |                 RichText::new(format!("cmd {:3.0}% / act {:3.0}%", self.throttle_target * 100.0, self.throttle
| 2985 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 2986 |                     .monospace() | support logic; impact depends on neighboring block and data flow. |
| 2987 |                     .color(Color32::YELLOW), | support logic; impact depends on neighboring block and data flo
| 2988 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2989 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 2990 |             // Brake | comment/documentation; changing affects understanding and intent traceability. |
| 2991 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 2992 |                 RichText::new("BRAKE") | support logic; impact depends on neighboring block and data flow. |
| 2993 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 2994 |                     .color(Color32::from_gray(145)), | support logic; impact depends on neighboring block and
| 2995 |             ); | support logic; impact depends on neighboring block and data flow. |
| 2996 |             let mut brk = self.brake_target; | support logic; impact depends on neighboring block and data flo
| 2997 |             let brk_resp = ui.add_enabled( | support logic; impact depends on neighboring block and data flow.
| 2998 |                 !ad_locked, | support logic; impact depends on neighboring block and data flow. |
| 2999 |                 Slider::new(&mut brk, 0.0..=1.0) | support logic; impact depends on neighboring block and data
| 3000 |                     .show_value(false) | support logic; impact depends on neighboring block and data flow. |
| 3001 |                     .trailing_fill(true), | support logic; impact depends on neighboring block and data flow.
```

```
| 3002 |             ); | support logic; impact depends on neighboring block and data flow. |
| 3003 |             if ad_locked { | runtime gate; condition changes can enable/disable critical paths. |
| 3004 |                 let _ = brk_resp.on_disabled_hover_text("Freio manual bloqueado enquanto o AD estiver ativo.")
| 3005 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 3006 |                 let brk_changed = (brk - self.brake_cmd_last).abs() > 0.01; | support logic; impact depends on
| 3007 |                 let brk_due = self.ticks.saturating_sub(self.brake_cmd_last_tick) >= 2; | support logic; impac
| 3008 |                 if brk != self.brake_target && (brk_changed \|\| brk_due) { | runtime gate; condition changes c
| 3009 |                     self.cmds.push(Cmd::SetBrake(brk)); | support logic; impact depends on neighboring block an
| 3010 |                     self.brake_cmd_last = brk; | support logic; impact depends on neighboring block and data fl
| 3011 |                     self.brake_cmd_last_tick = self.ticks; | support logic; impact depends on neighboring bloc
| 3012 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3013 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3014 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3015 |                 RichText::new(format!("cmd {:3.0}% / act {:3.0}%", self.brake_target * 100.0, self.brake * 100
| 3016 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 3017 |                     .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3018 |                     .color(Color32::RED), | support logic; impact depends on neighboring block and data flow.
| 3019 |             ); | support logic; impact depends on neighboring block and data flow. |
| 3020 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3021 |             // Direction | comment/documentation; changing affects understanding and intent traceability. |
| 3022 |             let dir_s = match self.bench.tcm.direction { | support logic; impact depends on neighboring block a
| 3023 |                 Direction::Forward => "F", | support logic; impact depends on neighboring block and data flow.
| 3024 |                 Direction::Reverse => "R", | support logic; impact depends on neighboring block and data flow.
| 3025 |                 Direction::Neutral => "N", | support logic; impact depends on neighboring block and data flow.
| 3026 |                 Direction::Park => "P", | support logic; impact depends on neighboring block and data flow. |
| 3027 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3028 |             let dir_resp = ui.add_enabled_ui(!ad_locked, \|ui\| direction_selector(ui, dir_s)); | support logic
| 3029 |             if ad_locked { | runtime gate; condition changes can enable/disable critical paths. |
| 3030 |                 let _ = dir_resp | support logic; impact depends on neighboring block and data flow. |
| 3031 |                     .response | support logic; impact depends on neighboring block and data flow. |
| 3032 |                     .on_disabled_hover_text("Direcao bloqueada enquanto o AD estiver ativo."); | support logic
| 3033 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3034 |             if let Some(k) = dir_resp.inner { | runtime gate; condition changes can enable/disable critical pa
| 3035 |                 match k { | branch dispatcher; arm changes alter behavior class handling. |
| 3036 |                     'F' => self.cmds.push(Cmd::SetDirection(Direction::Forward)), | support logic; impact depe
| 3037 |                     'R' => self.cmds.push(Cmd::SetDirection(Direction::Reverse)), | support logic; impact depe
| 3038 |                     'N' => self.cmds.push(Cmd::SetNeutral), | support logic; impact depends on neighboring bloc
| 3039 |                     _ => {} | support logic; impact depends on neighboring block and data flow. |
| 3040 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3041 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3042 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3043 |             // Auto-shift | comment/documentation; changing affects understanding and intent traceability. |
| 3044 |             let auto = self.bench.tcm.auto_mode == AutoShiftMode::Auto; | support logic; impact depends on neig
| 3045 |             let auto_btn = Button::new(RichText::new(if auto { "AUTO✓" } else { "MANUAL" }).size(11.0)) | supp
| 3046 |                 .fill(if auto { | support logic; impact depends on neighboring block and data flow. |
| 3047 |                     Color32::from_rgb(20, 85, 35) | support logic; impact depends on neighboring block and data
| 3048 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 3049 |                     Color32::from_gray(35) | support logic; impact depends on neighboring block and data flow.
| 3050 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3051 |             let auto_resp = ui.add_enabled(!ad_locked, auto_btn); | support logic; impact depends on neighborin
| 3052 |             if ad_locked { | runtime gate; condition changes can enable/disable critical paths. |
| 3053 |                 let _ = auto_resp.on_disabled_hover_text("Modo da transmissao bloqueado enquanto o AD estiver a
| 3054 |             } else if auto_resp.clicked() { | support logic; impact depends on neighboring block and data flow
| 3055 |                 self.cmds.push(Cmd::ToggleAutoShift); | support logic; impact depends on neighboring block and
| 3056 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3057 |             // PTO | comment/documentation; changing affects understanding and intent traceability. |
| 3058 |             let pto = self.bench.implement.pto_rear_enabled; | support logic; impact depends on neighboring bl
| 3059 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3060 |                 .add( | support logic; impact depends on neighboring block and data flow. |
| 3061 |                     Button::new(RichText::new(if pto { "PTO ON" } else { "PTO OFF" }).size(11.0)) | support lo
| 3062 |                         .fill(if pto { | support logic; impact depends on neighboring block and data flow. |
| 3063 |                             Color32::from_rgb(10, 95, 10) | support logic; impact depends on neighboring block
| 3064 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 3065 |                             Color32::from_gray(35) | support logic; impact depends on neighboring block and da
| 3066 |                         }), | support logic; impact depends on neighboring block and data flow. |
| 3067 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 3068 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3069 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3070 |                 self.cmds.push(if pto { | support logic; impact depends on neighboring block and data flow. |
| 3071 |                     Cmd::SetPtoMode(PtoMode::Off) | support logic; impact depends on neighboring block and data
| 3072 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 3073 |                     Cmd::SetPtoMode(PtoMode::Std540) | support logic; impact depends on neighboring block and
| 3074 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3075 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3076 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3077 |             // Lights | comment/documentation; changing affects understanding and intent traceability. |
```

```
| 3078 |             let wl = self.bench.bcm.work_lights_front; | support logic; impact depends on neighboring block and
| 3079 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3080 |                 .add( | support logic; impact depends on neighboring block and data flow. |
| 3081 |                     Button::new(RichText::new(if wl { "■ WRK" } else { "■ off" }).size(11.0)) | support logic;
| 3082 |                         .fill(if wl { | support logic; impact depends on neighboring block and data flow. |
| 3083 |                             Color32::from_rgb(85, 65, 8) | support logic; impact depends on neighboring block
| 3084 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 3085 |                             Color32::from_gray(32) | support logic; impact depends on neighboring block and da
| 3086 |                         }), | support logic; impact depends on neighboring block and data flow. |
| 3087 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 3088 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3089 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3090 |                 self.cmds.push(Cmd::ToggleWorkLights); | support logic; impact depends on neighboring block and
| 3091 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3092 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3093 |             // Pause / Reset | comment/documentation; changing affects understanding and intent traceability.
| 3094 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3095 |                 .add(Button::new(RichText::new("■ PAUSE").size(12.0)).fill(Color32::from_gray(42))) | support
| 3096 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3097 |             { /* toggle handled via bench.paused would need lib support; just use ticks */ | support logic; im
| 3098 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3099 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3100 |                 .add( | support logic; impact depends on neighboring block and data flow. |
| 3101 |                     Button::new(RichText::new("■ RESET").size(12.0)) | support logic; impact depends on neighb
| 3102 |                         .fill(Color32::from_rgb(70, 22, 22)), | support logic; impact depends on neighboring b
| 3103 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 3104 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3105 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3106 |                 self.cmds.push(Cmd::Reset); | support logic; impact depends on neighboring block and data flow
| 3107 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3108 |         }); | support logic; impact depends on neighboring block and data flow. |
| 3109 |         }); | support logic; impact depends on neighboring block and data flow. |
| 3110 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3111 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3112 | (blank) | blank separator; no runtime behavior. |
| 3113 | // ■■■
```

```
| 3154 |                 ui.group(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3155 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3156 |                         RichText::new("Leak Lab - Envelope correto") | support logic; impact depends on neighbo
| 3157 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 3158 |                             .color(Color32::from_rgb(95, 170, 255)) | support logic; impact depends on neighbor
| 3159 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 3160 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3161 |                     ui.label("Regra obrigatoria: min <= mean <= ideal <= max < rupture"); | support logic; impa
| 3162 |                     ui.label("Squeeze: tipico 10%~30%, Compression Set: menor tende a maior vida util."); | sup
| 3163 |                     ui.label("Base leak area: influencia vazao de fuga inicial (mm²)." ); | support logic; impa
| 3164 |                     ui.label("Rupture bar define limite de falha catastrofica e o risco de burst."); | support
| 3165 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3166 | (blank) | blank separator; no runtime behavior. |
| 3167 |                 ui.add_space(8.0); | support logic; impact depends on neighboring block and data flow. |
| 3168 |                 ui.group(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3169 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3170 |                         RichText::new("Leak Lab - Como operar sem erro") | support logic; impact depends on nei
| 3171 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 3172 |                             .color(Color32::from_rgb(95, 170, 255)) | support logic; impact depends on neighbor
| 3173 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 3174 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3175 |                     ui.label("1) Selecione circuito."); | support logic; impact depends on neighboring block an
| 3176 |                     ui.label("2) Ajuste parametros e valide (a UI bloqueia aplicacao invalida)."); | support lo
| 3177 |                     ui.label("3) APPLY MANUAL PARAMETERS."); | support logic; impact depends on neighboring blo
| 3178 |                     ui.label("4) RUN SCENARIO PREDICTION ou RUN MONTE CARLO."); | support logic; impact depends
| 3179 |                     ui.label("5) Leia ranking + timeline + plot + alertas."); | support logic; impact depends o
| 3180 |                     ui.label("6) Exporte CSV/JSON para rastreabilidade."); | support logic; impact depends on n
| 3181 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3182 | (blank) | blank separator; no runtime behavior. |
| 3183 |                 ui.add_space(8.0); | support logic; impact depends on neighboring block and data flow. |
| 3184 |                 ui.group(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3185 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3186 |                         RichText::new("Atalhos de diagnostico") | support logic; impact depends on neighboring
| 3187 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 3188 |                             .color(Color32::from_rgb(95, 170, 255)) | support logic; impact depends on neighbor
| 3189 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 3190 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3191 |                     ui.label("- EVENTS: filtre por severidade e subsystem."); | support logic; impact depends o
| 3192 |                     ui.label("- CAN: use Freeze antes de exportar snapshot."); | support logic; impact depends
| 3193 |                     ui.label("- UDS: siga Session -> Security -> Service -> Transfer Exit."); | protocol or I/O
| 3194 |                     ui.label("- ECM LIVE: Detect -> Connect -> Retrieve -> Export."); | support logic; impact o
| 3195 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3196 |             }); | support logic; impact depends on neighboring block and data flow. |
| 3197 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3198 | (blank) | blank separator; no runtime behavior. |
| 3199 |     fn tab_cluster(&self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flo
| 3200 |         let ecm = &self.bench.ecm; | support logic; impact depends on neighboring block and data flow. |
| 3201 |         let tcm = &self.bench.tcm; | support logic; impact depends on neighboring block and data flow. |
| 3202 |         let abs = &self.bench.abs; | support logic; impact depends on neighboring block and data flow. |
| 3203 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3204 |             // Left: big gauges + warning lamps | comment/documentation; changing affects understanding and int
| 3205 |             ui.vertical(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3206 |                 ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3207 |                     egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on n
| 3208 |                         arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3209 |                             ui, ecm.rpm, 0.0, 3000.0, "ENGINE", "RPM", 155.0, 2000.0, 2400.0, None, | support l
| 3210 |                         ) | support logic; impact depends on neighboring block and data flow. |
| 3211 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 3212 |                     egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on n
| 3213 |                         arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3214 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3215 |                             tcm.ground_speed_kmh, | support logic; impact depends on neighboring block and data
| 3216 |                             0.0, | support logic; impact depends on neighboring block and data flow. |
| 3217 |                             50.0, | support logic; impact depends on neighboring block and data flow. |
| 3218 |                             "SPEED", | support logic; impact depends on neighboring block and data flow. |
| 3219 |                             "km/h", | support logic; impact depends on neighboring block and data flow. |
| 3220 |                             155.0, | support logic; impact depends on neighboring block and data flow. |
| 3221 |                             40.0, | support logic; impact depends on neighboring block and data flow. |
| 3222 |                             48.0, | support logic; impact depends on neighboring block and data flow. |
| 3223 |                             None, | support logic; impact depends on neighboring block and data flow. |
| 3224 |                         ) | support logic; impact depends on neighboring block and data flow. |
| 3225 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 3226 |                     egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on n
| 3227 |                         arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3228 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3229 |                             ecm.coolant_temp_c, | support logic; impact depends on neighboring block and data
```

```
| 3230 |                                0.0, | support logic; impact depends on neighboring block and data flow. |
| 3231 |                                130.0, | support logic; impact depends on neighboring block and data flow. |
| 3232 |                                "COOLANT", | support logic; impact depends on neighboring block and data flow. |
| 3233 |                                "°C", | support logic; impact depends on neighboring block and data flow. |
| 3234 |                                125.0, | support logic; impact depends on neighboring block and data flow. |
| 3235 |                                100.0, | support logic; impact depends on neighboring block and data flow. |
| 3236 |                                108.0, | support logic; impact depends on neighboring block and data flow. |
| 3237 |                                None, | support logic; impact depends on neighboring block and data flow. |
| 3238 |                            ) | support logic; impact depends on neighboring block and data flow. |
| 3239 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 3240 |                        egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on n
| 3241 |                            arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3242 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 3243 |                                ecm.oil_pressure_kpa, | support logic; impact depends on neighboring block and data
| 3244 |                                0.0, | support logic; impact depends on neighboring block and data flow. |
| 3245 |                                700.0, | support logic; impact depends on neighboring block and data flow. |
| 3246 |                                "OIL PRESS", | support logic; impact depends on neighboring block and data flow. |
| 3247 |                                "kPa", | support logic; impact depends on neighboring block and data flow. |
| 3248 |                                125.0, | support logic; impact depends on neighboring block and data flow. |
| 3249 |                                0.0, | support logic; impact depends on neighboring block and data flow. |
| 3250 |                                0.0, | support logic; impact depends on neighboring block and data flow. |
| 3251 |                                Some(80.0), | support logic; impact depends on neighboring block and data flow. |
| 3252 |                            ) | support logic; impact depends on neighboring block and data flow. |
| 3253 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 3254 |                        egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on n
| 3255 |                            arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3256 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 3257 |                                ecm.boost_pressure_kpa, | support logic; impact depends on neighboring block and da
| 3258 |                                0.0, | support logic; impact depends on neighboring block and data flow. |
| 3259 |                                300.0, | support logic; impact depends on neighboring block and data flow. |
| 3260 |                                "BOOST", | support logic; impact depends on neighboring block and data flow. |
| 3261 |                                "kPa", | support logic; impact depends on neighboring block and data flow. |
| 3262 |                                110.0, | support logic; impact depends on neighboring block and data flow. |
| 3263 |                                250.0, | support logic; impact depends on neighboring block and data flow. |
| 3264 |                                280.0, | support logic; impact depends on neighboring block and data flow. |
| 3265 |                                None, | support logic; impact depends on neighboring block and data flow. |
| 3266 |                            ) | support logic; impact depends on neighboring block and data flow. |
| 3267 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 3268 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 3269 |                    egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data fl
| 3270 |                        .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data
| 3271 |                        .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3272 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3273 |                                RichText::new(" ■ WARNING CLUSTER") | support logic; impact depends on neighboring
| 3274 |                                    .size(11.0) | support logic; impact depends on neighboring block and data flow
| 3275 |                                    .color(Color32::from_gray(130)), | support logic; impact depends on neighboring
| 3276 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 3277 |                            ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block an
| 3278 |                                warning_lamp(ui, ecm.red_lamp, "RED STOP", Color32::RED); | support logic; impact
| 3279 |                                warning_lamp(ui, ecm.amber_lamp, "AMBER", Color32::GOLD); | support logic; impact
| 3280 |                                warning_lamp(ui, ecm.mil_active, "MIL", Color32::LIGHT_BLUE); | support logic; impa
| 3281 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3282 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3283 |                                    ecm.protect_lamp, | support logic; impact depends on neighboring block and data
| 3284 |                                    "PROTECT", | support logic; impact depends on neighboring block and data flow.
| 3285 |                                    Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring
| 3286 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3287 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3288 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3289 |                                    ecm.oil_pressure_kpa < 100.0 && ecm.rpm > 500.0, | support logic; impact depen
| 3290 |                                    "OIL PRESS", | support logic; impact depends on neighboring block and data flo
| 3291 |                                    Color32::RED, | support logic; impact depends on neighboring block and data fl
| 3292 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3293 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3294 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3295 |                                    ecm.coolant_temp_c > 105.0, | support logic; impact depends on neighboring bloc
| 3296 |                                    "COOLANT HI", | support logic; impact depends on neighboring block and data fl
| 3297 |                                    Color32::RED, | support logic; impact depends on neighboring block and data fl
| 3298 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3299 |                                warning_lamp(ui, ecm.def_level_pct < 10.0, "DEF LOW", Color32::YELLOW); | support
| 3300 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3301 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3302 |                                    ecm.dpf_soot_pct > 75.0, | support logic; impact depends on neighboring block a
| 3303 |                                    "DPF REGEN", | support logic; impact depends on neighboring block and data flo
| 3304 |                                    Color32::from_rgb(200, 80, 200), | support logic; impact depends on neighboring
| 3305 |                                ); | support logic; impact depends on neighboring block and data flow. |
```

```
| 3306 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3307 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3308 |                                    ecm.fuel_level_pct < 10.0, | support logic; impact depends on neighboring bloc
| 3309 |                                    "FUEL LOW", | support logic; impact depends on neighboring block and data flow
| 3310 |                                    Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 3311 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3312 |                                warning_lamp(ui, abs.abs_system_active, "ABS", Color32::WHITE); | support logic; i
| 3313 |                                warning_lamp(ui, abs.esp_system_active, "ESP", Color32::LIGHT_BLUE); | support log
| 3314 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3315 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3316 |                                    abs.tcs_system_active, | support logic; impact depends on neighboring block an
| 3317 |                                    "TCS", | support logic; impact depends on neighboring block and data flow. |
| 3318 |                                    Color32::from_rgb(80, 200, 255), | support logic; impact depends on neighborin
| 3319 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3320 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3321 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3322 |                                    self.bench.bcm.work_lights_front, | support logic; impact depends on neighbori
| 3323 |                                    "WORK LT", | support logic; impact depends on neighboring block and data flow.
| 3324 |                                    Color32::from_rgb(255, 215, 0), | support logic; impact depends on neighboring
| 3325 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3326 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3327 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3328 |                                    self.bench.implement.pto_rear_enabled, | support logic; impact depends on neig
| 3329 |                                    "PTO", | support logic; impact depends on neighboring block and data flow. |
| 3330 |                                    Color32::GREEN, | support logic; impact depends on neighboring block and data
| 3331 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3332 |                                warning_lamp(ui, self.bench.ad.engaged, "AD ON", Color32::GREEN); | support logic;
| 3333 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3334 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3335 |                                    self.bench.hcm.alarm_high_temp, | support logic; impact depends on neighboring
| 3336 |                                    "HYD TEMP", | support logic; impact depends on neighboring block and data flow
| 3337 |                                    Color32::RED, | support logic; impact depends on neighboring block and data fl
| 3338 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3339 |                                warning_lamp( | support logic; impact depends on neighboring block and data flow.
| 3340 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 3341 |                                    self.bench.hcm.filter_bypass_open, | support logic; impact depends on neighbor
| 3342 |                                    "HYD FILT", | support logic; impact depends on neighboring block and data flow
| 3343 |                                    Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 3344 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 3345 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 3346 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 3347 |                    egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data f
| 3348 |                        .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data
| 3349 |                        .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3350 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3351 |                                RichText::new("ENGINE") | support logic; impact depends on neighboring block and d
| 3352 |                                    .size(11.0) | support logic; impact depends on neighboring block and data flow
| 3353 |                                    .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on ne
| 3354 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 3355 |                            let w = 130.0; | support logic; impact depends on neighboring block and data flow. |
| 3356 |                            bar_gauge(ui, "Load", ecm.percent_load, 150.0, "%", w, Color32::YELLOW); | support log
| 3357 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3358 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 3359 |                                "Torque", | support logic; impact depends on neighboring block and data flow. |
| 3360 |                                ecm.actual_torque_nm, | support logic; impact depends on neighboring block and data
| 3361 |                                1200.0, | support logic; impact depends on neighboring block and data flow. |
| 3362 |                                "Nm", | support logic; impact depends on neighboring block and data flow. |
| 3363 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 3364 |                                Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighboring bl
| 3365 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 3366 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3367 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 3368 |                                "Power", | support logic; impact depends on neighboring block and data flow. |
| 3369 |                                ecm.power_kw(), | support logic; impact depends on neighboring block and data flow
| 3370 |                                260.0, | support logic; impact depends on neighboring block and data flow. |
| 3371 |                                "kW", | support logic; impact depends on neighboring block and data flow. |
| 3372 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 3373 |                                Color32::from_rgb(100, 220, 100), | support logic; impact depends on neighboring b
| 3374 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 3375 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3376 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 3377 |                                "Rail P", | support logic; impact depends on neighboring block and data flow. |
| 3378 |                                ecm.fuel_rail_pressure_mpa * 10.0, | support logic; impact depends on neighboring
| 3379 |                                2000.0, | support logic; impact depends on neighboring block and data flow. |
| 3380 |                                "bar", | support logic; impact depends on neighboring block and data flow. |
| 3381 |                                w, | support logic; impact depends on neighboring block and data flow. |
```

```
| 3382 |                            Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring bloc
| 3383 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 3384 |                        bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3385 |                            ui, | support logic; impact depends on neighboring block and data flow. |
| 3386 |                            "Exhaust", | support logic; impact depends on neighboring block and data flow. |
| 3387 |                            ecm.exhaust_temp_c, | support logic; impact depends on neighboring block and data f
| 3388 |                            900.0, | support logic; impact depends on neighboring block and data flow. |
| 3389 |                            "°C", | support logic; impact depends on neighboring block and data flow. |
| 3390 |                            w, | support logic; impact depends on neighboring block and data flow. |
| 3391 |                            Color32::from_rgb(255, 120, 30), | support logic; impact depends on neighboring blo
| 3392 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 3393 |                        bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3394 |                            ui, | support logic; impact depends on neighboring block and data flow. |
| 3395 |                            "Fuel/h", | support logic; impact depends on neighboring block and data flow. |
| 3396 |                            ecm.fuel_rate_lph, | support logic; impact depends on neighboring block and data fl
| 3397 |                            30.0, | support logic; impact depends on neighboring block and data flow. |
| 3398 |                            "L/h", | support logic; impact depends on neighboring block and data flow. |
| 3399 |                            w, | support logic; impact depends on neighboring block and data flow. |
| 3400 |                            Color32::YELLOW, | support logic; impact depends on neighboring block and data flow
| 3401 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 3402 |                        bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3403 |                            ui, | support logic; impact depends on neighboring block and data flow. |
| 3404 |                            "DPF", | support logic; impact depends on neighboring block and data flow. |
| 3405 |                            ecm.dpf_soot_pct, | support logic; impact depends on neighboring block and data flo
| 3406 |                            100.0, | support logic; impact depends on neighboring block and data flow. |
| 3407 |                            "%", | support logic; impact depends on neighboring block and data flow. |
| 3408 |                            w, | support logic; impact depends on neighboring block and data flow. |
| 3409 |                            if ecm.dpf_soot_pct > 75.0 { | runtime gate; condition changes can enable/disable c
| 3410 |                                Color32::RED | support logic; impact depends on neighboring block and data flow
| 3411 |                            } else { | support logic; impact depends on neighboring block and data flow. |
| 3412 |                                Color32::from_rgb(180, 100, 220) | support logic; impact depends on neighboring
| 3413 |                            }, | support logic; impact depends on neighboring block and data flow. |
| 3414 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 3415 |                        bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3416 |                            ui, | support logic; impact depends on neighboring block and data flow. |
| 3417 |                            "DEF", | support logic; impact depends on neighboring block and data flow. |
| 3418 |                            ecm.def_level_pct, | support logic; impact depends on neighboring block and data fl
| 3419 |                            100.0, | support logic; impact depends on neighboring block and data flow. |
| 3420 |                            "%", | support logic; impact depends on neighboring block and data flow. |
| 3421 |                            w, | support logic; impact depends on neighboring block and data flow. |
| 3422 |                            if ecm.def_level_pct < 10.0 { | runtime gate; condition changes can enable/disable
| 3423 |                                Color32::RED | support logic; impact depends on neighboring block and data flow
| 3424 |                            } else { | support logic; impact depends on neighboring block and data flow. |
| 3425 |                                Color32::from_rgb(0, 210, 210) | support logic; impact depends on neighboring b
| 3426 |                            }, | support logic; impact depends on neighboring block and data flow. |
| 3427 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 3428 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 3429 |                }); | support logic; impact depends on neighboring block and data flow. |
| 3430 |                ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3431 |                // Centre: transmission | comment/documentation; changing affects understanding and intent traceabi
| 3432 |                ui.vertical(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3433 |                    egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data fl
| 3434 |                        .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data
| 3435 |                        .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3436 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3437 |                            RichText::new("TRANSMISSION") | support logic; impact depends on neighboring block
| 3438 |                                .size(11.0) | support logic; impact depends on neighboring block and data flow
| 3439 |                                .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on ne
| 3440 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 3441 |                        let dc = match tcm.direction { | support logic; impact depends on neighboring block an
| 3442 |                            Direction::Forward => Color32::GREEN, | support logic; impact depends on neighborin
| 3443 |                            Direction::Reverse => Color32::RED, | support logic; impact depends on neighboring
| 3444 |                            Direction::Neutral => Color32::YELLOW, | support logic; impact depends on neighbor
| 3445 |                            Direction::Park => Color32::from_gray(180), | support logic; impact depends on nei
| 3446 |                        }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 3447 |                        let dl = match tcm.direction { | support logic; impact depends on neighboring block an
| 3448 |                            Direction::Forward => "■ FWD", | support logic; impact depends on neighboring bloc
| 3449 |                            Direction::Reverse => "■ REV", | support logic; impact depends on neighboring bloc
| 3450 |                            Direction::Neutral => "■ NEU", | support logic; impact depends on neighboring bloc
| 3451 |                            Direction::Park => "P", | support logic; impact depends on neighboring block and da
| 3452 |                        }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 3453 |                        ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data fl
| 3454 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3455 |                                RichText::new(&tcm.gear_label) | support logic; impact depends on neighboring b
| 3456 |                                    .size(38.0) | support logic; impact depends on neighboring block and data f
| 3457 |                                    .color(Color32::from_rgb(80, 220, 80)) | support logic; impact depends on n
```

```
| 3458 |                                 .strong(), | support logic; impact depends on neighboring block and data f
| 3459 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 3460 |                             ui.add_space(8.0); | support logic; impact depends on neighboring block and data f
| 3461 |                             ui.label(RichText::new(dl).size(22.0).color(dc).strong()); | support logic; impact
| 3462 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 3463 |                         let am = match tcm.auto_mode { | support logic; impact depends on neighboring block an
| 3464 |                             AutoShiftMode::Auto => "AUTO", | support logic; impact depends on neighboring bloc
| 3465 |                             AutoShiftMode::Manual => "MANUAL", | support logic; impact depends on neighboring b
| 3466 |                             AutoShiftMode::Hold => "HOLD", | support logic; impact depends on neighboring bloc
| 3467 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 3468 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data f
| 3469 |                             ui.label(RichText::new(am).size(12.0).color(Color32::YELLOW)); | support logic; imp
| 3470 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 3471 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3472 |                                 RichText::new(format!("Rng {}", tcm.range)) | support logic; impact depends on
| 3473 |                                     .size(12.0) | support logic; impact depends on neighboring block and data
| 3474 |                                     .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighborin
| 3475 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 3476 |                             if tcm.creeper_engaged { | runtime gate; condition changes can enable/disable crit
| 3477 |                                 ui.separator(); | support logic; impact depends on neighboring block and data
| 3478 |                                 ui.label(RichText::new("CREEP").size(11.0).color(Color32::YELLOW)); | support
| 3479 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 3480 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 3481 |                         let cc = match tcm.clutch_state { | support logic; impact depends on neighboring block
| 3482 |                             ClutchState::Locked => Color32::GREEN, | support logic; impact depends on neighbor
| 3483 |                             ClutchState::Modulating => Color32::YELLOW, | support logic; impact depends on nei
| 3484 |                             ClutchState::Filling => Color32::from_rgb(0, 220, 220), | support logic; impact de
| 3485 |                             ClutchState::Open => Color32::from_gray(70), | support logic; impact depends on ne
| 3486 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 3487 |                         digital_readout(ui, "Clutch", &format!("{}", tcm.clutch_state), cc); | support logic;
| 3488 |                         bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3489 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3490 |                             "Slip", | support logic; impact depends on neighboring block and data flow. |
| 3491 |                             tcm.clutch_slip_pct, | support logic; impact depends on neighboring block and data
| 3492 |                             100.0, | support logic; impact depends on neighboring block and data flow. |
| 3493 |                             "%", | support logic; impact depends on neighboring block and data flow. |
| 3494 |                             120.0, | support logic; impact depends on neighboring block and data flow. |
| 3495 |                             if tcm.clutch_slip_pct > 20.0 { | runtime gate; condition changes can enable/disab
| 3496 |                                 Color32::YELLOW | support logic; impact depends on neighboring block and data
| 3497 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 3498 |                                 Color32::GREEN | support logic; impact depends on neighboring block and data f
| 3499 |                             }, | support logic; impact depends on neighboring block and data flow. |
| 3500 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3501 |                         if tcm.is_shifting { | runtime gate; condition changes can enable/disable critical pat
| 3502 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3503 |                                 RichText::new("■■ SHIFTING…") | support logic; impact depends on neighboring b
| 3504 |                                     .size(11.0) | support logic; impact depends on neighboring block and data
| 3505 |                                     .color(Color32::YELLOW), | support logic; impact depends on neighboring bl
| 3506 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 3507 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 3508 |                         digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 3509 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3510 |                             "Speed", | support logic; impact depends on neighboring block and data flow. |
| 3511 |                             &format!("{:.2} km/h", tcm.ground_speed_kmh), | support logic; impact depends on n
| 3512 |                             Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and data
| 3513 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3514 |                         digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 3515 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3516 |                             "Out RPM", | support logic; impact depends on neighboring block and data flow. |
| 3517 |                             &format!("{:.0}", tcm.output_shaft_rpm), | support logic; impact depends on neighb
| 3518 |                             Color32::from_gray(200), | support logic; impact depends on neighboring block and
| 3519 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3520 |                         digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 3521 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3522 |                             "Shifts", | support logic; impact depends on neighboring block and data flow. |
| 3523 |                             &format!("{}", tcm.total_shifts), | support logic; impact depends on neighboring b
| 3524 |                             Color32::from_gray(160), | support logic; impact depends on neighboring block and
| 3525 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3526 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 3527 |                 egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data f
| 3528 |                     .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data
| 3529 |                     .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3530 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3531 |                             RichText::new("ABS / WHEEL SPEEDS") | support logic; impact depends on neighboring
| 3532 |                                 .size(11.0) | support logic; impact depends on neighboring block and data flow
| 3533 |                                 .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on ne
```

```
| 3534 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3535 |                         for (i, w) in abs.wheels.iter().enumerate() { | iteration logic; may alter timing, thre
| 3536 |                             let n = ["FL", "FR", "RL", "RR"][i]; | support logic; impact depends on neighboring
| 3537 |                             let c = if w.abs_active { | support logic; impact depends on neighboring block and
| 3538 |                                 Color32::RED | support logic; impact depends on neighboring block and data flow
| 3539 |                             } else if w.tcs_active { | support logic; impact depends on neighboring block and d
| 3540 |                                 Color32::from_rgb(80, 200, 255) | support logic; impact depends on neighboring
| 3541 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 3542 |                                 Color32::GREEN | support logic; impact depends on neighboring block and data fl
| 3543 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 3544 |                             ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and dat
| 3545 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 3546 |                                     RichText::new(format!( | support logic; impact depends on neighboring block
| 3547 |                                         "{}: {:5.1}km/h {}", | support logic; impact depends on neighboring bl
| 3548 |                                         n, w.speed, w.valve_state | support logic; impact depends on neighborin
| 3549 |                                     )) | support logic; impact depends on neighboring block and data flow. |
| 3550 |                                     .size(10.5) | support logic; impact depends on neighboring block and data f
| 3551 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 3552 |                                     .color(c), | support logic; impact depends on neighboring block and data fl
| 3553 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 3554 |                                 if w.slip_ratio > 0.1 { | runtime gate; condition changes can enable/disable co
| 3555 |                                     ui.label( | support logic; impact depends on neighboring block and data flo
| 3556 |                                         RichText::new(format!("slip:{:.0}%", w.slip_ratio * 100.0)) | support l
| 3557 |                                             .size(10.0) | support logic; impact depends on neighboring block an
| 3558 |                                             .color(Color32::YELLOW), | support logic; impact depends on neighbo
| 3559 |                                     ); | support logic; impact depends on neighboring block and data flow. |
| 3560 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 3561 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 3562 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 3563 |                         let ec = match abs.esp_condition { | support logic; impact depends on neighboring block
| 3564 |                             EspCondition::Neutral => Color32::GREEN, | support logic; impact depends on neighbo
| 3565 |                             EspCondition::Understeer => Color32::YELLOW, | support logic; impact depends on ne
| 3566 |                             _ => Color32::RED, | support logic; impact depends on neighboring block and data fl
| 3567 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 3568 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data fl
| 3569 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3570 |                                 RichText::new("ESP:") | support logic; impact depends on neighboring block and
| 3571 |                                     .size(11.0) | support logic; impact depends on neighboring block and data f
| 3572 |                                     .color(Color32::from_gray(140)), | support logic; impact depends on neighbo
| 3573 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 3574 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3575 |                                 RichText::new(format!("{}", abs.esp_condition)) | support logic; impact depends
| 3576 |                                     .size(11.0) | support logic; impact depends on neighboring block and data f
| 3577 |                                     .color(ec) | support logic; impact depends on neighboring block and data fl
| 3578 |                                     .strong(), | support logic; impact depends on neighboring block and data fl
| 3579 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 3580 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 3581 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 3582 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3583 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3584 |                 // Right: fuel/def/battery + hitch | comment/documentation; changing affects understanding and inte
| 3585 |                 ui.vertical(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3586 |                     egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on neigh
| 3587 |                         arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3588 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3589 |                             ecm.fuel_level_pct, | support logic; impact depends on neighboring block and data flow
| 3590 |                             0.0, | support logic; impact depends on neighboring block and data flow. |
| 3591 |                             100.0, | support logic; impact depends on neighboring block and data flow. |
| 3592 |                             "FUEL", | support logic; impact depends on neighboring block and data flow. |
| 3593 |                             "%", | support logic; impact depends on neighboring block and data flow. |
| 3594 |                             125.0, | support logic; impact depends on neighboring block and data flow. |
| 3595 |                             0.0, | support logic; impact depends on neighboring block and data flow. |
| 3596 |                             0.0, | support logic; impact depends on neighboring block and data flow. |
| 3597 |                             Some(12.0), | support logic; impact depends on neighboring block and data flow. |
| 3598 |                         ) | support logic; impact depends on neighboring block and data flow. |
| 3599 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 3600 |                     egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on neigh
| 3601 |                         arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3602 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3603 |                             ecm.def_level_pct, | support logic; impact depends on neighboring block and data flow.
| 3604 |                             0.0, | support logic; impact depends on neighboring block and data flow. |
| 3605 |                             100.0, | support logic; impact depends on neighboring block and data flow. |
| 3606 |                             "DEF/AdBlue", | support logic; impact depends on neighboring block and data flow. |
| 3607 |                             "%", | support logic; impact depends on neighboring block and data flow. |
| 3608 |                             125.0, | support logic; impact depends on neighboring block and data flow. |
| 3609 |                             0.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 3610 |                         0.0, | support logic; impact depends on neighboring block and data flow. |
| 3611 |                         Some(10.0), | support logic; impact depends on neighboring block and data flow. |
| 3612 |                     ) | support logic; impact depends on neighboring block and data flow. |
| 3613 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3614 |                 egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on neigh
| 3615 |                     arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 3616 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 3617 |                         self.bench.bcm.battery_voltage, | support logic; impact depends on neighboring block an
| 3618 |                         9.0, | support logic; impact depends on neighboring block and data flow. |
| 3619 |                         16.0, | support logic; impact depends on neighboring block and data flow. |
| 3620 |                         "BATTERY", | support logic; impact depends on neighboring block and data flow. |
| 3621 |                         "V", | support logic; impact depends on neighboring block and data flow. |
| 3622 |                         105.0, | support logic; impact depends on neighboring block and data flow. |
| 3623 |                         0.0, | support logic; impact depends on neighboring block and data flow. |
| 3624 |                         0.0, | support logic; impact depends on neighboring block and data flow. |
| 3625 |                         Some(11.5), | support logic; impact depends on neighboring block and data flow. |
| 3626 |                     ) | support logic; impact depends on neighboring block and data flow. |
| 3627 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3628 |                 egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data f
| 3629 |                     .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data
| 3630 |                     .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3631 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3632 |                             RichText::new("SERVICE") | support logic; impact depends on neighboring block and
| 3633 |                                 .size(11.0) | support logic; impact depends on neighboring block and data flow
| 3634 |                                 .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on ne
| 3635 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3636 |                         digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 3637 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3638 |                             "Hours", | support logic; impact depends on neighboring block and data flow. |
| 3639 |                             &format!("{:.1} h", ecm.engine_hours), | support logic; impact depends on neighbor
| 3640 |                             Color32::from_gray(200), | support logic; impact depends on neighboring block and
| 3641 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3642 |                         digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 3643 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3644 |                             "Svc due", | support logic; impact depends on neighboring block and data flow. |
| 3645 |                             &format!("{:.1} h", ecm.service_hours_left), | support logic; impact depends on ne
| 3646 |                             if ecm.service_hours_left < 50.0 { | runtime gate; condition changes can enable/di
| 3647 |                                 Color32::RED | support logic; impact depends on neighboring block and data flo
| 3648 |                             } else if ecm.service_hours_left < 100.0 { | support logic; impact depends on neigh
| 3649 |                                 Color32::YELLOW | support logic; impact depends on neighboring block and data
| 3650 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 3651 |                                 Color32::GREEN | support logic; impact depends on neighboring block and data f
| 3652 |                             }, | support logic; impact depends on neighboring block and data flow. |
| 3653 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3654 |                         digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 3655 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 3656 |                             "NOx tail", | support logic; impact depends on neighboring block and data flow. |
| 3657 |                             &format!("{:.0} ppm", ecm.nox_tailpipe_ppm), | support logic; impact depends on ne
| 3658 |                             if ecm.nox_tailpipe_ppm > 500.0 { | runtime gate; condition changes can enable/dis
| 3659 |                                 Color32::RED | support logic; impact depends on neighboring block and data flo
| 3660 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 3661 |                                 Color32::from_gray(180) | support logic; impact depends on neighboring block a
| 3662 |                             }, | support logic; impact depends on neighboring block and data flow. |
| 3663 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3664 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 3665 |             }); | support logic; impact depends on neighboring block and data flow. |
| 3666 |         }); | support logic; impact depends on neighboring block and data flow. |
| 3667 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3668 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3669 | (blank) | blank separator; no runtime behavior. |
| 3670 | // ■■■
```

```
| 3686 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3687 |             let nc = if self.can_mode == CanMode::Network { | support logic; impact depends on neighboring bloc
| 3688 |                 Color32::from_rgb(45, 105, 210) | support logic; impact depends on neighboring block and data f
| 3689 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 3690 |                 Color32::from_gray(35) | support logic; impact depends on neighboring block and data flow. |
| 3691 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3692 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3693 |                 .add(Button::new(RichText::new("■ Signals").size(12.0)).fill(mc)) | support logic; impact depe
| 3694 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3695 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3696 |                 self.can_mode = CanMode::Signals; | support logic; impact depends on neighboring block and data
| 3697 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3698 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3699 |                 .add(Button::new(RichText::new("■ Trace").size(12.0)).fill(tc)) | support logic; impact depend
| 3700 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3701 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3702 |                 self.can_mode = CanMode::Trace; | support logic; impact depends on neighboring block and data f
| 3703 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3704 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3705 |                 .add(Button::new(RichText::new("■ Network").size(12.0)).fill(nc)) | support logic; impact depe
| 3706 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3707 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3708 |                 self.can_mode = CanMode::Network; | support logic; impact depends on neighboring block and data
| 3709 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3710 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3711 |             let fc = if self.can_freeze { | support logic; impact depends on neighboring block and data flow.
| 3712 |                 Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 3713 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 3714 |                 Color32::from_gray(38) | support logic; impact depends on neighboring block and data flow. |
| 3715 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3716 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 3717 |                 .add( | support logic; impact depends on neighboring block and data flow. |
| 3718 |                     Button::new( | support logic; impact depends on neighboring block and data flow. |
| 3719 |                         RichText::new(if self.can_freeze { | support logic; impact depends on neighboring bloc
| 3720 |                             "■ FROZEN" | support logic; impact depends on neighboring block and data flow. |
| 3721 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 3722 |                             "■ Freeze" | support logic; impact depends on neighboring block and data flow. |
| 3723 |                         }) | support logic; impact depends on neighboring block and data flow. |
| 3724 |                         .size(12.0), | support logic; impact depends on neighboring block and data flow. |
| 3725 |                     ) | support logic; impact depends on neighboring block and data flow. |
| 3726 |                     .fill(fc), | support logic; impact depends on neighboring block and data flow. |
| 3727 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 3728 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 3729 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3730 |                 self.can_freeze = !self.can_freeze; | support logic; impact depends on neighboring block and da
| 3731 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3732 |             if ui.button("■ Clear").clicked() { | runtime gate; condition changes can enable/disable critical
| 3733 |                 self.sig_map.clear(); | support logic; impact depends on neighboring block and data flow. |
| 3734 |                 self.trace_snap.clear(); | support logic; impact depends on neighboring block and data flow. |
| 3735 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3736 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3737 |             ui.label(RichText::new("Filter:").size(11.0).color(Color32::GRAY)); | support logic; impact depends
| 3738 |             ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 3739 |                 [220.0, 20.0], | support logic; impact depends on neighboring block and data flow. |
| 3740 |                 TextEdit::singleline(&mut self.can_filter) | support logic; impact depends on neighboring bloc
| 3741 |                     .hint_text("text \| sa:00 \| pgn:61444 \| id:18F00400"), | support logic; impact depends o
| 3742 |             ); | support logic; impact depends on neighboring block and data flow. |
| 3743 |             ui.checkbox(&mut self.can_hide_stale, "Hide stale"); | support logic; impact depends on neighboring
| 3744 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3745 |             ui.label(RichText::new("Rows").size(10.8).color(Color32::from_gray(145))); | support logic; impact
| 3746 |             ui.add(Slider::new(&mut self.can_signal_limit, 50..=1200).text("sig")); | support logic; impact dep
| 3747 |             ui.add(Slider::new(&mut self.can_trace_limit, 100..=2000).text("trace")); | support logic; impact d
| 3748 |             if matches!(self.can_mode, CanMode::Trace) { | runtime gate; condition changes can enable/disable c
| 3749 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3750 |                 ui.label(RichText::new("Trace tick").size(10.8).color(Color32::from_gray(145))); | support logi
| 3751 |                 ui.add( | support logic; impact depends on neighboring block and data flow. |
| 3752 |                     Slider::new(&mut self.can_trace_update_every_ticks, 1..=30) | support logic; impact depends
| 3753 |                         .text("every") | support logic; impact depends on neighboring block and data flow. |
| 3754 |                         .suffix("t"), | support logic; impact depends on neighboring block and data flow. |
| 3755 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 3756 |                 ui.checkbox(&mut self.can_trace_pause_on_scroll, "Pause on scroll"); | support logic; impact de
| 3757 |                 if self.can_trace_pause_ticks_left > 0 { | runtime gate; condition changes can enable/disable c
| 3758 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3759 |                         RichText::new(format!("paused {}t", self.can_trace_pause_ticks_left)) | support logic;
| 3760 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3761 |                             .color(Color32::YELLOW), | support logic; impact depends on neighboring block and c
```

```
| 3762 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3763 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3764 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3765 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3766 |             let gw = &self.bench.gateway; | support logic; impact depends on neighboring block and data flow.
| 3767 |             let state_col = match gw.bus_state { | support logic; impact depends on neighboring block and data
| 3768 |                 BusState::ErrorActive => Color32::GREEN, | support logic; impact depends on neighboring block a
| 3769 |                 BusState::ErrorPassive => Color32::YELLOW, | support logic; impact depends on neighboring bloc
| 3770 |                 BusState::BusOff => Color32::RED, | support logic; impact depends on neighboring block and data
| 3771 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3772 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3773 |                 RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 3774 |                     "Bus:{} Load:{:.0}% Frames:{} {}/s Err:{}", | support logic; impact depends on neighboring
| 3775 |                     gw.bus_state, gw.bus_load_pct, gw.total_tx, gw.bus.fps, gw.total_errors | support logic; i
| 3776 |                 )) | support logic; impact depends on neighboring block and data flow. |
| 3777 |                 .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 3778 |                 .color(state_col), | support logic; impact depends on neighboring block and data flow. |
| 3779 |             ); | support logic; impact depends on neighboring block and data flow. |
| 3780 |         }); | support logic; impact depends on neighboring block and data flow. |
| 3781 |         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3782 |             RichText::new("Tip: use sa:XX / pgn:NNNN / id:XXXXXXXX for deterministic filtering") | support log
| 3783 |                 .size(9.8) | support logic; impact depends on neighboring block and data flow. |
| 3784 |                 .color(Color32::from_gray(125)), | support logic; impact depends on neighboring block and data
| 3785 |         ); | support logic; impact depends on neighboring block and data flow. |
| 3786 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3787 |         match self.can_mode { | branch dispatcher; arm changes alter behavior class handling. |
| 3788 |             CanMode::Signals => self.can_signals(ui), | support logic; impact depends on neighboring block and
| 3789 |             CanMode::Trace => self.can_trace(ui), | support logic; impact depends on neighboring block and data
| 3790 |             CanMode::Network => self.can_network_overview(ui), | support logic; impact depends on neighboring
| 3791 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3792 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3793 | (blank) | blank separator; no runtime behavior. |
| 3794 |     fn can_network_overview(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior a
| 3795 |         let net = &self.bench.can_net; | support logic; impact depends on neighboring block and data flow. |
| 3796 |         let health = (net.network_health_score_01() * 100.0).round(); | support logic; impact depends on neigh
| 3797 |         let hc = if health >= 85.0 { | support logic; impact depends on neighboring block and data flow. |
| 3798 |             Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 3799 |         } else if health >= 65.0 { | support logic; impact depends on neighboring block and data flow. |
| 3800 |             Color32::YELLOW | support logic; impact depends on neighboring block and data flow. |
| 3801 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 3802 |             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 3803 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3804 |         ui.label(RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 3805 |             "Multi-bus overview \| Health: {:.0}% \| Frames(all buses): {} \| Nodes online: {} \| Errors(total
| 3806 |             health, | support logic; impact depends on neighboring block and data flow. |
| 3807 |             net.total_frames_all_buses, | support logic; impact depends on neighboring block and data flow. |
| 3808 |             net.online_count(), | support logic; impact depends on neighboring block and data flow. |
| 3809 |             net.total_errors() | support logic; impact depends on neighboring block and data flow. |
| 3810 |         )).size(11.0).color(hc)); | support logic; impact depends on neighboring block and data flow. |
| 3811 |         if !self.can_note.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 3812 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3813 |                 RichText::new(&self.can_note) | support logic; impact depends on neighboring block and data fl
| 3814 |                     .size(10.4) | support logic; impact depends on neighboring block and data flow. |
| 3815 |                     .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighboring block and data
| 3816 |             ); | support logic; impact depends on neighboring block and data flow. |
| 3817 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3818 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3819 |             const BUS_NAMES: [&str; 5] = ["Powertrain", "Chassis", "Body", "ISOBUS", "Diagnostic"]; | support
| 3820 |             ComboBox::from_id_salt("can::target_bus") | support logic; impact depends on neighboring block and
| 3821 |                 .selected_text(BUS_NAMES[self.can_bus_idx.min(4)]) | support logic; impact depends on neighbor
| 3822 |                 .show_ui(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3823 |                     for (i, n) in BUS_NAMES.iter().enumerate() { | iteration logic; may alter timing, throughp
| 3824 |                         ui.selectable_value(&mut self.can_bus_idx, i, *n); | support logic; impact depends on
| 3825 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3826 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3827 | (blank) | blank separator; no runtime behavior. |
| 3828 |             if ui.button("Inject Bit").clicked() { | runtime gate; condition changes can enable/disable critic
| 3829 |                 self.cmds.push(Cmd::CanInjectBitError(self.can_bus_idx)); | support logic; impact depends on n
| 3830 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3831 |             if ui.button("Inject Ack").clicked() { | runtime gate; condition changes can enable/disable critic
| 3832 |                 self.cmds.push(Cmd::CanInjectAckError(self.can_bus_idx)); | support logic; impact depends on n
| 3833 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3834 |             if ui.button("Inject BusOff").clicked() { | runtime gate; condition changes can enable/disable cri
| 3835 |                 self.cmds.push(Cmd::CanInjectBusOff(self.can_bus_idx)); | support logic; impact depends on nei
| 3836 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3837 |             if ui.button("Inject Babbling").clicked() { | runtime gate; condition changes can enable/disable c
```

```
| 3838 |                 self.cmds.push(Cmd::CanInjectBabbling(self.can_bus_idx)); | support logic; impact depends on ne
| 3839 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3840 |             if ui.button("Clear Bus").clicked() { | runtime gate; condition changes can enable/disable critical
| 3841 |                 self.cmds.push(Cmd::CanClearBusInjections(self.can_bus_idx)); | support logic; impact depends o
| 3842 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3843 |             if ui.button("Clear All").clicked() { | runtime gate; condition changes can enable/disable critical
| 3844 |                 self.cmds.push(Cmd::CanClearAllInjections); | support logic; impact depends on neighboring bloc
| 3845 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3846 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3847 |             if ui.button("Export CSV").clicked() { | runtime gate; condition changes can enable/disable critica
| 3848 |                 let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring b
| 3849 |                 let name = format!("can_network_snapshot_{}.csv", ts); | support logic; impact depends on neigh
| 3850 |                 if let Some(path) = Self::pick_save_path(&name, "csv") { | runtime gate; condition changes can
| 3851 |                     self.cmds.push(Cmd::CanExportSnapshotCsv(path)); | support logic; impact depends on neighbo
| 3852 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3853 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3854 |             if ui.button("Export JSON").clicked() { | runtime gate; condition changes can enable/disable critic
| 3855 |                 let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring b
| 3856 |                 let name = format!("can_network_snapshot_{}.json", ts); | support logic; impact depends on neig
| 3857 |                 if let Some(path) = Self::pick_save_path(&name, "json") { | runtime gate; condition changes can
| 3858 |                     self.cmds.push(Cmd::CanExportSnapshotJson(path)); | support logic; impact depends on neighb
| 3859 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3860 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3861 |         }); | support logic; impact depends on neighboring block and data flow. |
| 3862 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3863 | (blank) | blank separator; no runtime behavior. |
| 3864 |         let bus_cards = [ | support logic; impact depends on neighboring block and data flow. |
| 3865 |             ("HS-CAN Powertrain", &net.powertrain), | support logic; impact depends on neighboring block and da
| 3866 |             ("HS-CAN Chassis", &net.chassis), | support logic; impact depends on neighboring block and data flo
| 3867 |             ("MS-CAN Body", &net.body), | support logic; impact depends on neighboring block and data flow. |
| 3868 |             ("ISOBUS", &net.isobus), | support logic; impact depends on neighboring block and data flow. |
| 3869 |             ("Diag", &net.diagnostic), | support logic; impact depends on neighboring block and data flow. |
| 3870 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 3871 | (blank) | blank separator; no runtime behavior. |
| 3872 |         egui::Grid::new("can::net_buses") | support logic; impact depends on neighboring block and data flow.
| 3873 |             .num_columns(6) | support logic; impact depends on neighboring block and data flow. |
| 3874 |             .striped(true) | support logic; impact depends on neighboring block and data flow. |
| 3875 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 3876 |                 for h in ["BUS", "STATE", "LOAD", "TX", "LOG", "ERRORS"] { | iteration logic; may alter timing,
| 3877 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3878 |                         RichText::new(h) | support logic; impact depends on neighboring block and data flow. |
| 3879 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3880 |                             .color(Color32::from_gray(160)) | support logic; impact depends on neighboring bloc
| 3881 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 3882 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3883 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3884 |                 ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 3885 | (blank) | blank separator; no runtime behavior. |
| 3886 |                 for (name, bus) in bus_cards { | iteration logic; may alter timing, throughput, and safety cons
| 3887 |                     let state_col = match bus.state { | support logic; impact depends on neighboring block and
| 3888 |                         auto_breaking::can_network::BusState::ErrorActive => Color32::GREEN, | support logic; i
| 3889 |                         auto_breaking::can_network::BusState::ErrorPassive => Color32::YELLOW, | support logic
| 3890 |                         auto_breaking::can_network::BusState::BusOff => Color32::RED, | support logic; impact o
| 3891 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3892 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3893 |                         RichText::new(name) | support logic; impact depends on neighboring block and data flow
| 3894 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3895 |                             .color(Color32::from_gray(220)), | support logic; impact depends on neighboring blo
| 3896 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3897 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3898 |                         RichText::new(format!("{}", bus.state)) | support logic; impact depends on neighboring
| 3899 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3900 |                             .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3901 |                             .color(state_col), | support logic; impact depends on neighboring block and data fl
| 3902 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3903 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3904 |                         RichText::new(format!("{:.1}%", bus.bus_load_pct)) | support logic; impact depends on n
| 3905 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3906 |                             .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3907 |                             .color(Color32::from_rgb(100, 205, 255)), | support logic; impact depends on neighb
| 3908 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3909 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3910 |                         RichText::new(format!("{}", bus.total_tx)) | support logic; impact depends on neighbori
| 3911 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3912 |                             .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3913 |                             .color(Color32::from_gray(190)), | support logic; impact depends on neighboring blo
```

```
| 3914 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3915 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3916 |                         RichText::new(format!("{:.0}", bus.frame_log.len())) | support logic; impact depends on
| 3917 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3918 |                             .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3919 |                             .color(Color32::from_gray(160)), | support logic; impact depends on neighboring bl
| 3920 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3921 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3922 |                         RichText::new(format!("{}", bus.total_errors)) | support logic; impact depends on neigh
| 3923 |                             .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 3924 |                             .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3925 |                             .color(if bus.total_errors > 0 { | support logic; impact depends on neighboring bl
| 3926 |                                 Color32::YELLOW | support logic; impact depends on neighboring block and data
| 3927 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 3928 |                                 Color32::GREEN | support logic; impact depends on neighboring block and data f
| 3929 |                             }), | support logic; impact depends on neighboring block and data flow. |
| 3930 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3931 |                     ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 3932 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3933 |             }); | support logic; impact depends on neighboring block and data flow. |
| 3934 | (blank) | blank separator; no runtime behavior. |
| 3935 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3936 |         ui.columns(2, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 3937 |             egui::Frame::group(cols[0].style()) | support logic; impact depends on neighboring block and data
| 3938 |                 .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data fl
| 3939 |                 .show(&mut cols[0], \|ui\| { | support logic; impact depends on neighboring block and data flo
| 3940 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3941 |                         RichText::new("VCM ROUTING TABLE") | support logic; impact depends on neighboring bloc
| 3942 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 3943 |                             .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighb
| 3944 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3945 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3946 |                     for (pgn_id, src, dst) in &net.routing { | iteration logic; may alter timing, throughput,
| 3947 |                         let pgn_name = auto_breaking::find_pgn(*pgn_id) | support logic; impact depends on nei
| 3948 |                             .map(\|p\| p.name) | support logic; impact depends on neighboring block and data f
| 3949 |                             .unwrap_or("???"); | error propagation point; changes alter failure propagation se
| 3950 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3951 |                             RichText::new(format!( | support logic; impact depends on neighboring block and da
| 3952 |                                 "PGN {:5} {:8}  {} -> {}", | support logic; impact depends on neighboring bloc
| 3953 |                                 pgn_id, pgn_name, src, dst | support logic; impact depends on neighboring bloc
| 3954 |                             )) | support logic; impact depends on neighboring block and data flow. |
| 3955 |                             .size(10.2) | support logic; impact depends on neighboring block and data flow. |
| 3956 |                             .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3957 |                             .color(Color32::from_gray(200)), | support logic; impact depends on neighboring bl
| 3958 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3959 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3960 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3961 | (blank) | blank separator; no runtime behavior. |
| 3962 |             egui::Frame::group(cols[1].style()) | support logic; impact depends on neighboring block and data
| 3963 |                 .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data fl
| 3964 |                 .show(&mut cols[1], \|ui\| { | support logic; impact depends on neighboring block and data flo
| 3965 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3966 |                         RichText::new("LATEST NETWORK ERRORS") | support logic; impact depends on neighboring
| 3967 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 3968 |                             .color(Color32::from_rgb(255, 150, 80)), | support logic; impact depends on neighb
| 3969 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 3970 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 3971 |                     for err in net.all_errors().take(14) { | iteration logic; may alter timing, throughput, an
| 3972 |                         let sa = err.source_sa.map_or("--".into(), \|v\| format!("{:02X}", v)); | support logi
| 3973 |                         let id = err | support logic; impact depends on neighboring block and data flow. |
| 3974 |                             .raw_id | support logic; impact depends on neighboring block and data flow. |
| 3975 |                             .map_or("--------".into(), \|v\| format!("{:08X}", v)); | support logic; impact de
| 3976 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3977 |                             RichText::new(format!( | support logic; impact depends on neighboring block and da
| 3978 |                                 "{:.2}s [{}] SA {} ID {} {:?}", | error propagation point; changes alter failu
| 3979 |                                 err.timestamp, err.bus, sa, id, err.kind | support logic; impact depends on ne
| 3980 |                             )) | support logic; impact depends on neighboring block and data flow. |
| 3981 |                             .size(9.9) | support logic; impact depends on neighboring block and data flow. |
| 3982 |                             .monospace() | support logic; impact depends on neighboring block and data flow. |
| 3983 |                             .color(Color32::from_rgb(255, 170, 110)), | support logic; impact depends on neigh
| 3984 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 3985 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 3986 |                             RichText::new(err.description) | support logic; impact depends on neighboring bloc
| 3987 |                                 .size(9.6) | support logic; impact depends on neighboring block and data flow.
| 3988 |                                 .color(Color32::from_gray(150)), | support logic; impact depends on neighboring
| 3989 |                         ); | support logic; impact depends on neighboring block and data flow. |
```

```
| 3990 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3991 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 3992 |         }); | support logic; impact depends on neighboring block and data flow. |
| 3993 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 3994 | (blank) | blank separator; no runtime behavior. |
| 3995 |     fn can_signals(&self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flo
| 3996 |         let filt = self.can_filter.clone(); | support logic; impact depends on neighboring block and data flow
| 3997 |         // Table headers | comment/documentation; changing affects understanding and intent traceability. |
| 3998 |         egui::Grid::new("can::sig_hdr") | support logic; impact depends on neighboring block and data flow. |
| 3999 |             .num_columns(6) | support logic; impact depends on neighboring block and data flow. |
| 4000 |             .min_col_width(60.0) | support logic; impact depends on neighboring block and data flow. |
| 4001 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4002 |                 for h in ["AGE(s)", "PGN", "SA", "PERIOD", "COUNT", "DECODED VALUES"] { | iteration logic; may
| 4003 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4004 |                         RichText::new(h) | support logic; impact depends on neighboring block and data flow. |
| 4005 |                             .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 4006 |                             .color(Color32::from_gray(155)) | support logic; impact depends on neighboring bloc
| 4007 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 4008 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 4009 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4010 |                 ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 4011 |             }); | support logic; impact depends on neighboring block and data flow. |
| 4012 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4013 |         let t = self.bench.elapsed; | support logic; impact depends on neighboring block and data flow. |
| 4014 |         let mut sigs: Vec<(&(u32, u8), &Signal)> = self | support logic; impact depends on neighboring block a
| 4015 |             .sig_map | support logic; impact depends on neighboring block and data flow. |
| 4016 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 4017 |             .filter(\|((pgn, sa), s)\| Self::can_filter_match_signal(&filt, *pgn, *sa, s)) | support logic; imp
| 4018 |             .filter(\|(_, s)\| !self.can_hide_stale \|\| s.fresh) | support logic; impact depends on neighborin
| 4019 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 4020 |         sigs.sort_by_key(\|((pgn, sa), _)\| (*sa, *pgn)); | support logic; impact depends on neighboring block
| 4021 | (blank) | blank separator; no runtime behavior. |
| 4022 |         let total = sigs.len(); | support logic; impact depends on neighboring block and data flow. |
| 4023 |         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4024 |             RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 4025 |                 "Showing {} / {} signals", | support logic; impact depends on neighboring block and data flow.
| 4026 |                 total.min(self.can_signal_limit), | support logic; impact depends on neighboring block and data
| 4027 |                 total | support logic; impact depends on neighboring block and data flow. |
| 4028 |             )) | support logic; impact depends on neighboring block and data flow. |
| 4029 |             .size(10.2) | support logic; impact depends on neighboring block and data flow. |
| 4030 |             .color(Color32::from_gray(130)), | support logic; impact depends on neighboring block and data flow
| 4031 |         ); | support logic; impact depends on neighboring block and data flow. |
| 4032 | (blank) | blank separator; no runtime behavior. |
| 4033 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 4034 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 4035 |             .max_height(ui.available_height()) | support logic; impact depends on neighboring block and data f
| 4036 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4037 |                 for ((pgn, sa), sig) in sigs.iter().take(self.can_signal_limit) { | iteration logic; may alter
| 4038 |                     let age = t - sig.last_ts; | support logic; impact depends on neighboring block and data f
| 4039 |                     let ac = if age < 0.05 { | support logic; impact depends on neighboring block and data flo
| 4040 |                         Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 4041 |                     } else if age < 0.5 { | support logic; impact depends on neighboring block and data flow.
| 4042 |                         Color32::from_gray(220) | support logic; impact depends on neighboring block and data
| 4043 |                     } else if age < 2.0 { | support logic; impact depends on neighboring block and data flow.
| 4044 |                         Color32::from_gray(150) | support logic; impact depends on neighboring block and data
| 4045 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 4046 |                         Color32::from_gray(80) | support logic; impact depends on neighboring block and data f
| 4047 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4048 |                     let vals = sig | support logic; impact depends on neighboring block and data flow. |
| 4049 |                         .decoded | support logic; impact depends on neighboring block and data flow. |
| 4050 |                         .iter() | support logic; impact depends on neighboring block and data flow. |
| 4051 |                         .take(4) | support logic; impact depends on neighboring block and data flow. |
| 4052 |                         .map(\|(n, v, u)\| { | support logic; impact depends on neighboring block and data flo
| 4053 |                             format!( | support logic; impact depends on neighboring block and data flow. |
| 4054 |                                 "{}:{:.1}{}", | support logic; impact depends on neighboring block and data fl
| 4055 |                                 n.split_whitespace().next().unwrap_or("?"), | error propagation point; changes
| 4056 |                                 v, | support logic; impact depends on neighboring block and data flow. |
| 4057 |                                 u | support logic; impact depends on neighboring block and data flow. |
| 4058 |                             ) | support logic; impact depends on neighboring block and data flow. |
| 4059 |                         }) | support logic; impact depends on neighboring block and data flow. |
| 4060 |                         .collect::<Vec<_>>() | support logic; impact depends on neighboring block and data flo
| 4061 |                         .join("  ") | support logic; impact depends on neighboring block and data flow. |
| 4062 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 4063 |                         ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4064 |                             [56.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4065 |                             Label::new( | support logic; impact depends on neighboring block and data flow. |
```

```
| 4066 |                                RichText::new(format!("{:>5.2}", age)) | support logic; impact depends on neigh
| 4067 |                                    .size(10.2) | support logic; impact depends on neighboring block and data f
| 4068 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4069 |                                    .color(ac), | support logic; impact depends on neighboring block and data f
| 4070 |                            ), | support logic; impact depends on neighboring block and data flow. |
| 4071 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 4072 |                        ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4073 |                            [190.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4074 |                            Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4075 |                                RichText::new(format!("{:>5} {}", pgn, sig.pgn_name)) | support logic; impact d
| 4076 |                                    .size(10.2) | support logic; impact depends on neighboring block and data f
| 4077 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4078 |                                    .color(Color32::GOLD), | support logic; impact depends on neighboring bloc
| 4079 |                            ), | support logic; impact depends on neighboring block and data flow. |
| 4080 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 4081 |                        ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4082 |                            [125.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4083 |                            Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4084 |                                RichText::new(format!("0x{:02X} {}", sa, sig.sa_name)) | support logic; impact
| 4085 |                                    .size(10.2) | support logic; impact depends on neighboring block and data f
| 4086 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4087 |                                    .color(Color32::from_rgb(120, 200, 255)), | support logic; impact depends d
| 4088 |                            ), | support logic; impact depends on neighboring block and data flow. |
| 4089 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 4090 |                        ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4091 |                            [76.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4092 |                            Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4093 |                                RichText::new(format!("{:>5.0}ms", sig.period_ms)) | support logic; impact depe
| 4094 |                                    .size(10.2) | support logic; impact depends on neighboring block and data f
| 4095 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4096 |                                    .color(Color32::from_gray(150)), | support logic; impact depends on neighbo
| 4097 |                            ), | support logic; impact depends on neighboring block and data flow. |
| 4098 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 4099 |                        ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4100 |                            [74.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4101 |                            Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4102 |                                RichText::new(format!("{:>7}", sig.count)) | support logic; impact depends on n
| 4103 |                                    .size(10.2) | support logic; impact depends on neighboring block and data f
| 4104 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4105 |                                    .color(Color32::from_gray(130)), | support logic; impact depends on neighbo
| 4106 |                            ), | support logic; impact depends on neighboring block and data flow. |
| 4107 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 4108 |                        ui.label(RichText::new(&vals).size(10.2).color(ac)); | support logic; impact depends on
| 4109 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 4110 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4111 |            }); | support logic; impact depends on neighboring block and data flow. |
| 4112 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4113 | (blank) | blank separator; no runtime behavior. |
| 4114 |     fn can_trace(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-
| 4115 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4116 |             ui.label(RichText::new("Trace View:").size(10.8).color(Color32::GRAY)); | support logic; impact dep
| 4117 |             ui.selectable_value(&mut self.can_trace_tree, false, "Table"); | support logic; impact depends on n
| 4118 |             ui.selectable_value(&mut self.can_trace_tree, true, "Tree"); | support logic; impact depends on ne
| 4119 |         }); | support logic; impact depends on neighboring block and data flow. |
| 4120 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4121 | (blank) | blank separator; no runtime behavior. |
| 4122 |         let filtered_rows: Vec<&TraceRow> = self | support logic; impact depends on neighboring block and data
| 4123 |             .trace_snap | support logic; impact depends on neighboring block and data flow. |
| 4124 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 4125 |             .filter(\|r\| Self::can_filter_match_trace(&self.can_filter, r)) | support logic; impact depends on
| 4126 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 4127 |         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4128 |             RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 4129 |                 "Rows after filter: {} (render limit {})", | support logic; impact depends on neighboring bloc
| 4130 |                 filtered_rows.len(), | support logic; impact depends on neighboring block and data flow. |
| 4131 |                 self.can_trace_limit | support logic; impact depends on neighboring block and data flow. |
| 4132 |             )) | support logic; impact depends on neighboring block and data flow. |
| 4133 |             .size(10.2) | support logic; impact depends on neighboring block and data flow. |
| 4134 |             .color(Color32::from_gray(130)), | support logic; impact depends on neighboring block and data flo
| 4135 |         ); | support logic; impact depends on neighboring block and data flow. |
| 4136 | (blank) | blank separator; no runtime behavior. |
| 4137 |         if self.can_trace_tree { | runtime gate; condition changes can enable/disable critical paths. |
| 4138 |             let mut by_sa: BTreeMap<u8, Vec<TraceRow>> = BTreeMap::new(); | support logic; impact depends on n
| 4139 |             for row in filtered_rows.iter().rev().take(self.can_trace_limit) { | iteration logic; may alter ti
| 4140 |                 by_sa.entry(row.2).or_default().push((*row).clone()); | support logic; impact depends on neigh
| 4141 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 4142 | (blank) | blank separator; no runtime behavior. |
| 4143 |             ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 4144 |                 .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flo
| 4145 |                 .max_height(ui.available_height()) | support logic; impact depends on neighboring block and da
| 4146 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4147 |                     for (sa, rows) in by_sa { | iteration logic; may alter timing, throughput, and safety cons
| 4148 |                         CollapsingHeader::new(format!("Node SA 0x{:02X} ({} frames)", sa, rows.len())) | suppo
| 4149 |                             .default_open(sa == 0x00 \|\| sa == 0x03) | support logic; impact depends on neigh
| 4150 |                             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data f
| 4151 |                                 let mut by_pgn: BTreeMap<String, Vec<TraceRow>> = BTreeMap::new(); | support l
| 4152 |                                 for r in rows { | iteration logic; may alter timing, throughput, and safety co
| 4153 |                                     by_pgn.entry(r.5.clone()).or_default().push(r); | support logic; impact de
| 4154 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 4155 |                                 for (pgn, mut frames) in by_pgn { | iteration logic; may alter timing, through
| 4156 |                                     frames.sort_by(\|a, b\| a.0.partial_cmp(&b.0).unwrap_or(std::cmp::Ordering
| 4157 |                                     CollapsingHeader::new(format!("{} ({})", pgn, frames.len())) | support log
| 4158 |                                         .default_open(false) | support logic; impact depends on neighboring bl
| 4159 |                                         .show(ui, \|ui\| { | support logic; impact depends on neighboring bloc
| 4160 |                                             for (ts, raw_id, _sa, dlc, hex, _pgn_sa, decoded) in | iteration l
| 4161 |                                                 frames.iter().rev().take(12).rev() | support logic; impact dep
| 4162 |                                             { | scope delimiter; structural only, but wrong placement changes
| 4163 |                                                 ui.horizontal_wrapped(\|ui\| { | support logic; impact depends
| 4164 |                                                     ui.label( | support logic; impact depends on neighboring b
| 4165 |                                                         RichText::new(format!("t={:8.3}s", ts)) | support logi
| 4166 |                                                             .monospace() | support logic; impact depends on ne
| 4167 |                                                             .size(10.0) | support logic; impact depends on nei
| 4168 |                                                             .color(Color32::from_gray(130)), | support logic;
| 4169 |                                                     ); | support logic; impact depends on neighboring block an
| 4170 |                                                     ui.label( | support logic; impact depends on neighboring b
| 4171 |                                                         RichText::new(format!("ID={:08X}", raw_id)) | support
| 4172 |                                                             .monospace() | support logic; impact depends on ne
| 4173 |                                                             .size(10.0) | support logic; impact depends on nei
| 4174 |                                                             .color(Color32::LIGHT_BLUE), | support logic; impac
| 4175 |                                                     ); | support logic; impact depends on neighboring block an
| 4176 |                                                     ui.label( | support logic; impact depends on neighboring b
| 4177 |                                                         RichText::new(format!("DLC={}", dlc)) | support logic;
| 4178 |                                                             .monospace() | support logic; impact depends on ne
| 4179 |                                                             .size(10.0) | support logic; impact depends on nei
| 4180 |                                                             .color(Color32::from_gray(140)), | support logic;
| 4181 |                                                     ); | support logic; impact depends on neighboring block an
| 4182 |                                                     ui.label( | support logic; impact depends on neighboring b
| 4183 |                                                         RichText::new(hex) | support logic; impact depends on
| 4184 |                                                             .monospace() | support logic; impact depends on ne
| 4185 |                                                             .size(10.0) | support logic; impact depends on nei
| 4186 |                                                             .color(Color32::from_gray(200)), | support logic;
| 4187 |                                                     ); | support logic; impact depends on neighboring block an
| 4188 |                                                     if !decoded.is_empty() { | runtime gate; condition changes
| 4189 |                                                         ui.label( | support logic; impact depends on neighbori
| 4190 |                                                             RichText::new(decoded) | support logic; impact dep
| 4191 |                                                                 .size(10.0) | support logic; impact depends on
| 4192 |                                                                 .color(Color32::from_gray(190)), | support log
| 4193 |                                                         ); | support logic; impact depends on neighboring bloc
| 4194 |                                                     } | scope delimiter; structural only, but wrong placement
| 4195 |                                                 }); | support logic; impact depends on neighboring block and d
| 4196 |                                             } | scope delimiter; structural only, but wrong placement changes
| 4197 |                                         }); | support logic; impact depends on neighboring block and data flow
| 4198 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 4199 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 4200 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4201 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 4202 |             return; | support logic; impact depends on neighboring block and data flow. |
| 4203 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4204 | (blank) | blank separator; no runtime behavior. |
| 4205 |         egui::Grid::new("can::trace_hdr") | support logic; impact depends on neighboring block and data flow.
| 4206 |             .num_columns(6) | support logic; impact depends on neighboring block and data flow. |
| 4207 |             .min_col_width(50.0) | support logic; impact depends on neighboring block and data flow. |
| 4208 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4209 |                 for h in ["TIME(s)", "CAN-ID", "DLC", "HEX DATA", "PGN", "DECODED"] { | iteration logic; may a
| 4210 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4211 |                         RichText::new(h) | support logic; impact depends on neighboring block and data flow. |
| 4212 |                             .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 4213 |                             .color(Color32::from_gray(155)) | support logic; impact depends on neighboring bloc
| 4214 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 4215 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 4216 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4217 |                 ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
```

```
| 4218 |             }); | support logic; impact depends on neighboring block and data flow. |
| 4219 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4220 |         // stick_to_bottom only when live; when frozen the user can scroll freely | comment/documentation; char
| 4221 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 4222 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 4223 |             .max_height(ui.available_height()) | support logic; impact depends on neighboring block and data f
| 4224 |             .stick_to_bottom(!self.can_freeze) | support logic; impact depends on neighboring block and data f
| 4225 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4226 |                 for row in filtered_rows.iter().rev().take(self.can_trace_limit).rev() { | iteration logic; may
| 4227 |                     let (ts, raw_id, sa, dlc, hex, pgn_sa, decoded) = *row; | support logic; impact depends on
| 4228 |                     let sc = match *sa { | support logic; impact depends on neighboring block and data flow. |
| 4229 |                         0x00 => Color32::from_rgb(80, 220, 80), | support logic; impact depends on neighboring
| 4230 |                         0x03 => Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring
| 4231 |                         0x27 => Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and d
| 4232 |                         0x1E => Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 4233 |                         0x0B => Color32::WHITE, | support logic; impact depends on neighboring block and data
| 4234 |                         _ => Color32::from_gray(175), | support logic; impact depends on neighboring block and
| 4235 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4236 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 4237 |                         ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4238 |                             [68.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4239 |                             Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4240 |                                 RichText::new(format!("{:8.3}", ts)) | support logic; impact depends on neighbo
| 4241 |                                     .size(10.0) | support logic; impact depends on neighboring block and data
| 4242 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 4243 |                                     .color(Color32::from_gray(110)), | support logic; impact depends on neighbo
| 4244 |                             ), | support logic; impact depends on neighboring block and data flow. |
| 4245 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 4246 |                         ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4247 |                             [84.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4248 |                             Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4249 |                                 RichText::new(format!("{:08X}", raw_id)) | support logic; impact depends on nei
| 4250 |                                     .size(10.0) | support logic; impact depends on neighboring block and data
| 4251 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 4252 |                                     .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighboring
| 4253 |                             ), | support logic; impact depends on neighboring block and data flow. |
| 4254 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 4255 |                         ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4256 |                             [32.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4257 |                             Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4258 |                                 RichText::new(format!("{}", dlc)) | support logic; impact depends on neighborin
| 4259 |                                     .size(10.0) | support logic; impact depends on neighboring block and data
| 4260 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 4261 |                                     .color(Color32::from_gray(130)), | support logic; impact depends on neighbo
| 4262 |                             ), | support logic; impact depends on neighboring block and data flow. |
| 4263 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 4264 |                         ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4265 |                             [205.0, 18.0], | support logic; impact depends on neighboring block and data flow.
| 4266 |                             Label::new( | support logic; impact depends on neighboring block and data flow. |
| 4267 |                                 RichText::new(hex) | support logic; impact depends on neighboring block and da
| 4268 |                                     .size(10.0) | support logic; impact depends on neighboring block and data
| 4269 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 4270 |                                     .color(Color32::from_gray(195)), | support logic; impact depends on neighbo
| 4271 |                             ), | support logic; impact depends on neighboring block and data flow. |
| 4272 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 4273 |                         ui.add_sized([180.0, 18.0], Label::new(RichText::new(pgn_sa).size(10.0).color(sc))); |
| 4274 |                         ui.label(RichText::new(decoded).size(10.0).color(Color32::from_gray(200))); | support
| 4275 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 4276 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4277 |             }); | support logic; impact depends on neighboring block and data flow. |
| 4278 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4279 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4280 | (blank) | blank separator; no runtime behavior. |
| 4281 | // ■■■
```

```
| 4294 |                     Button::new( | support logic; impact depends on neighboring block and data flow. |
| 4295 |                         RichText::new(if self.ev_pause { | support logic; impact depends on neighboring block a
| 4296 |                             "■ FROZEN" | support logic; impact depends on neighboring block and data flow. |
| 4297 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 4298 |                             "■ Pause" | support logic; impact depends on neighboring block and data flow. |
| 4299 |                         }) | support logic; impact depends on neighboring block and data flow. |
| 4300 |                         .size(12.0), | support logic; impact depends on neighboring block and data flow. |
| 4301 |                     ) | support logic; impact depends on neighboring block and data flow. |
| 4302 |                     .fill(pc), | support logic; impact depends on neighboring block and data flow. |
| 4303 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 4304 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 4305 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4306 |                 self.ev_pause = !self.ev_pause; | support logic; impact depends on neighboring block and data f
| 4307 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4308 |             if ui.button("■ Clear").clicked() { | runtime gate; condition changes can enable/disable critical
| 4309 |                 self.events.clear(); | support logic; impact depends on neighboring block and data flow. |
| 4310 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4311 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4312 |             ui.label(RichText::new("Filter:").size(11.0).color(Color32::GRAY)); | support logic; impact depends
| 4313 |             ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 4314 |                 [150.0, 20.0], | support logic; impact depends on neighboring block and data flow. |
| 4315 |                 TextEdit::singleline(&mut self.ev_filter).hint_text("text filter…"), | support logic; impact de
| 4316 |             ); | support logic; impact depends on neighboring block and data flow. |
| 4317 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4318 |             // Level filter buttons | comment/documentation; changing affects understanding and intent traceabi
| 4319 |             for (lvl, lbl, col) in [ | iteration logic; may alter timing, throughput, and safety constraints.
| 4320 |                 (EventLevel::Debug, "DBG", Color32::from_gray(140)), | support logic; impact depends on neighbo
| 4321 |                 (EventLevel::Info, "INFO", Color32::LIGHT_BLUE), | support logic; impact depends on neighboring
| 4322 |                 (EventLevel::Ok, "OK", Color32::GREEN), | support logic; impact depends on neighboring block an
| 4323 |                 (EventLevel::Warn, "WARN", Color32::YELLOW), | support logic; impact depends on neighboring blo
| 4324 |                 (EventLevel::Critical, "CRIT", Color32::RED), | support logic; impact depends on neighboring bl
| 4325 |             ] { | support logic; impact depends on neighboring block and data flow. |
| 4326 |                 let sel = self.ev_min_level == lvl; | support logic; impact depends on neighboring block and da
| 4327 |                 let fill = if sel { | support logic; impact depends on neighboring block and data flow. |
| 4328 |                     Color32::from_gray(60) | support logic; impact depends on neighboring block and data flow.
| 4329 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 4330 |                     Color32::from_gray(30) | support logic; impact depends on neighboring block and data flow.
| 4331 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4332 |                 if ui | runtime gate; condition changes can enable/disable critical paths. |
| 4333 |                     .add(Button::new(RichText::new(lbl).size(11.0).color(col)).fill(fill)) | support logic; imp
| 4334 |                     .clicked() | support logic; impact depends on neighboring block and data flow. |
| 4335 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4336 |                     self.ev_min_level = lvl; | support logic; impact depends on neighboring block and data flow
| 4337 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4338 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4339 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4340 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4341 |                 RichText::new(format!("{} events", self.events.len())) | support logic; impact depends on neigh
| 4342 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 4343 |                     .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block and d
| 4344 |             ); | support logic; impact depends on neighboring block and data flow. |
| 4345 |         }); | support logic; impact depends on neighboring block and data flow. |
| 4346 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4347 | (blank) | blank separator; no runtime behavior. |
| 4348 |         // Column headers | comment/documentation; changing affects understanding and intent traceability. |
| 4349 |         egui::Grid::new("events::hdr") | support logic; impact depends on neighboring block and data flow. |
| 4350 |             .num_columns(4) | support logic; impact depends on neighboring block and data flow. |
| 4351 |             .min_col_width(40.0) | support logic; impact depends on neighboring block and data flow. |
| 4352 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4353 |                 for h in ["TIME(s)", "LVL", "SOURCE", "EVENT"] { | iteration logic; may alter timing, throughpu
| 4354 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4355 |                         RichText::new(h) | support logic; impact depends on neighboring block and data flow. |
| 4356 |                             .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 4357 |                             .color(Color32::from_gray(150)) | support logic; impact depends on neighboring bloc
| 4358 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 4359 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 4360 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4361 |                 ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 4362 |             }); | support logic; impact depends on neighboring block and data flow. |
| 4363 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4364 | (blank) | blank separator; no runtime behavior. |
| 4365 |         let filt = self.ev_filter.to_lowercase(); | support logic; impact depends on neighboring block and data
| 4366 |         let min_lvl = self.ev_min_level; | support logic; impact depends on neighboring block and data flow. |
| 4367 |         let level_order = \|l: &EventLevel\| -> u8 { | support logic; impact depends on neighboring block and d
| 4368 |             match l { | branch dispatcher; arm changes alter behavior class handling. |
| 4369 |                 EventLevel::Debug => 0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 4370 |                 EventLevel::Info => 1, | support logic; impact depends on neighboring block and data flow. |
| 4371 |                 EventLevel::Ok => 2, | support logic; impact depends on neighboring block and data flow. |
| 4372 |                 EventLevel::Warn => 3, | support logic; impact depends on neighboring block and data flow. |
| 4373 |                 EventLevel::Critical => 4, | support logic; impact depends on neighboring block and data flow.
| 4374 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4375 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4376 |         let min_ord = level_order(&min_lvl); | support logic; impact depends on neighboring block and data flo
| 4377 | (blank) | blank separator; no runtime behavior. |
| 4378 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 4379 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 4380 |             .stick_to_bottom(!self.ev_pause) | support logic; impact depends on neighboring block and data flo
| 4381 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4382 |                 for (idx, ev) in self | iteration logic; may alter timing, throughput, and safety constraints.
| 4383 |                     .events | support logic; impact depends on neighboring block and data flow. |
| 4384 |                     .iter() | support logic; impact depends on neighboring block and data flow. |
| 4385 |                     .rev() | support logic; impact depends on neighboring block and data flow. |
| 4386 |                     .filter(\|e\| level_order(&e.lvl) >= min_ord) | support logic; impact depends on neighborin
| 4387 |                     .filter(\|e\| { | support logic; impact depends on neighboring block and data flow. |
| 4388 |                         filt.is_empty() | support logic; impact depends on neighboring block and data flow. |
| 4389 |                             \|\| e.msg.to_lowercase().contains(&filt) | support logic; impact depends on neigh
| 4390 |                             \|\| e.source.to_lowercase().contains(&filt) | support logic; impact depends on ne
| 4391 |                     }) | support logic; impact depends on neighboring block and data flow. |
| 4392 |                     .take(300) | support logic; impact depends on neighboring block and data flow. |
| 4393 |                     .enumerate() | support logic; impact depends on neighboring block and data flow. |
| 4394 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4395 |                     let (icon, col) = match ev.lvl { | support logic; impact depends on neighboring block and
| 4396 |                         EventLevel::Debug => ("·", Color32::from_gray(100)), | support logic; impact depends o
| 4397 |                         EventLevel::Info => ("■", Color32::LIGHT_BLUE), | support logic; impact depends on nei
| 4398 |                         EventLevel::Ok => ("✓", Color32::GREEN), | support logic; impact depends on neighborin
| 4399 |                         EventLevel::Warn => ("■", Color32::YELLOW), | support logic; impact depends on neighbo
| 4400 |                         EventLevel::Critical => ("✗", Color32::RED), | support logic; impact depends on neighbo
| 4401 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4402 |                     ui.push_id(format!("events::row_{}", idx), \|ui\| { | support logic; impact depends on neig
| 4403 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data f
| 4404 |                             ui.add_sized( | support logic; impact depends on neighboring block and data flow.
| 4405 |                                 [56.0, 18.0], | support logic; impact depends on neighboring block and data flo
| 4406 |                                 Label::new( | support logic; impact depends on neighboring block and data flow
| 4407 |                                     RichText::new(format!("{:7.2}", ev.ts)) | support logic; impact depends on
| 4408 |                                         .size(10.5) | support logic; impact depends on neighboring block and da
| 4409 |                                         .monospace() | support logic; impact depends on neighboring block and
| 4410 |                                         .color(Color32::from_gray(115)), | support logic; impact depends on nei
| 4411 |                                 ), | support logic; impact depends on neighboring block and data flow. |
| 4412 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4413 |                             ui.add_sized( | support logic; impact depends on neighboring block and data flow.
| 4414 |                                 [20.0, 18.0], | support logic; impact depends on neighboring block and data flo
| 4415 |                                 Label::new(RichText::new(icon.to_string()).size(11.0).color(col)), | support lo
| 4416 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4417 |                             ui.add_sized( | support logic; impact depends on neighboring block and data flow.
| 4418 |                                 [78.0, 18.0], | support logic; impact depends on neighboring block and data flo
| 4419 |                                 Label::new( | support logic; impact depends on neighboring block and data flow
| 4420 |                                     RichText::new(format!("[{:<5}]", ev.source)) | support logic; impact depend
| 4421 |                                         .size(10.5) | support logic; impact depends on neighboring block and da
| 4422 |                                         .monospace() | support logic; impact depends on neighboring block and
| 4423 |                                         .color(Color32::from_gray(165)), | support logic; impact depends on nei
| 4424 |                                 ), | support logic; impact depends on neighboring block and data flow. |
| 4425 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4426 |                             ui.label(RichText::new(&ev.msg).size(10.5).color(col)); | support logic; impact dep
| 4427 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 4428 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 4429 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4430 |             }); | support logic; impact depends on neighboring block and data flow. |
| 4431 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4432 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4433 | (blank) | blank separator; no runtime behavior. |
| 4434 | // ■■■
```

```
| 4446 |             .count(); | support logic; impact depends on neighboring block and data flow. |
| 4447 |         let total = self.bench.boot.ecus.len(); | support logic; impact depends on neighboring block and data f
| 4448 |         let unhealthy_nodes = gw | support logic; impact depends on neighboring block and data flow. |
| 4449 |             .nodes | support logic; impact depends on neighboring block and data flow. |
| 4450 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 4451 |             .filter(\|n\| n.tec > 0 \|\| n.rec > 0 \|\| !matches!(n.state, BusState::ErrorActive)) | support lo
| 4452 |             .count(); | support logic; impact depends on neighboring block and data flow. |
| 4453 |         let net_health = if matches!(gw.bus_state, BusState::BusOff) { | support logic; impact depends on neigh
| 4454 |             "CRITICAL" | support logic; impact depends on neighboring block and data flow. |
| 4455 |         } else if unhealthy_nodes > 0 \|\| gw.total_errors > 0 { | support logic; impact depends on neighboring
| 4456 |             "DEGRADED" | support logic; impact depends on neighboring block and data flow. |
| 4457 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 4458 |             "HEALTHY" | support logic; impact depends on neighboring block and data flow. |
| 4459 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4460 |         let net_col = match net_health { | support logic; impact depends on neighboring block and data flow. |
| 4461 |             "CRITICAL" => Color32::RED, | support logic; impact depends on neighboring block and data flow. |
| 4462 |             "DEGRADED" => Color32::YELLOW, | support logic; impact depends on neighboring block and data flow.
| 4463 |             _ => Color32::GREEN, | support logic; impact depends on neighboring block and data flow. |
| 4464 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4465 |         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4466 |             RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 4467 |                 "J1939 HS-CAN 500kbps \| Bus:{} \| Nodes:{}/{} \| TotalFrames:{}", | support logic; impact depe
| 4468 |                 gw.bus_state, | support logic; impact depends on neighboring block and data flow. |
| 4469 |                 online, | support logic; impact depends on neighboring block and data flow. |
| 4470 |                 total, | support logic; impact depends on neighboring block and data flow. |
| 4471 |                 gw.total_tx | support logic; impact depends on neighboring block and data flow. |
| 4472 |             )) | support logic; impact depends on neighboring block and data flow. |
| 4473 |             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 4474 |             .color(Color32::from_gray(185)), | support logic; impact depends on neighboring block and data flow
| 4475 |         ); | support logic; impact depends on neighboring block and data flow. |
| 4476 |         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4477 |             RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 4478 |                 "Network health: {} \| Unhealthy nodes: {} \| Total errors: {}", | support logic; impact depend
| 4479 |                 net_health, unhealthy_nodes, gw.total_errors | support logic; impact depends on neighboring bl
| 4480 |             )) | support logic; impact depends on neighboring block and data flow. |
| 4481 |             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 4482 |             .color(net_col), | support logic; impact depends on neighboring block and data flow. |
| 4483 |         ); | support logic; impact depends on neighboring block and data flow. |
| 4484 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4485 | (blank) | blank separator; no runtime behavior. |
| 4486 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 4487 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 4488 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4489 |                 egui::Grid::new("ecu_net::grid") | support logic; impact depends on neighboring block and data
| 4490 |                     .num_columns(7) | support logic; impact depends on neighboring block and data flow. |
| 4491 |                     .striped(true) | support logic; impact depends on neighboring block and data flow. |
| 4492 |                     .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4493 |                         for h in [ | iteration logic; may alter timing, throughput, and safety constraints. |
| 4494 |                             "MODULE", | support logic; impact depends on neighboring block and data flow. |
| 4495 |                             "SA", | support logic; impact depends on neighboring block and data flow. |
| 4496 |                             "STAGE", | support logic; impact depends on neighboring block and data flow. |
| 4497 |                             "ONLINE AT", | support logic; impact depends on neighboring block and data flow. |
| 4498 |                             "TEC", | support logic; impact depends on neighboring block and data flow. |
| 4499 |                             "REC", | support logic; impact depends on neighboring block and data flow. |
| 4500 |                             "BUS STATE", | support logic; impact depends on neighboring block and data flow. |
| 4501 |                         ] { | support logic; impact depends on neighboring block and data flow. |
| 4502 |                             ui.label(RichText::new(h).size(11.0).color(Color32::from_gray(150))); | support log
| 4503 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 4504 |                         ui.end_row(); | support logic; impact depends on neighboring block and data flow. |
| 4505 |                         for ecu in &self.bench.boot.ecus { | iteration logic; may alter timing, throughput, and
| 4506 |                             let sc = match ecu.stage { | support logic; impact depends on neighboring block and
| 4507 |                                 EcuBootStage::Running => Color32::GREEN, | support logic; impact depends on ne
| 4508 |                                 EcuBootStage::Fault => Color32::RED, | support logic; impact depends on neighb
| 4509 |                                 EcuBootStage::Unpowered => Color32::from_gray(55), | support logic; impact depe
| 4510 |                                 EcuBootStage::AddressClaiming => Color32::YELLOW, | support logic; impact depe
| 4511 |                                 _ => Color32::from_gray(185), | support logic; impact depends on neighboring b
| 4512 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 4513 |                             let node = gw.nodes.iter().find(\|n\| n.source_addr == ecu.sa); | support logic; i
| 4514 |                             let (tec, rec, bs) = node.map_or((0u16, 0u16, "N/A"), \|n\| { | support logic; impa
| 4515 |                                 ( | support logic; impact depends on neighboring block and data flow. |
| 4516 |                                     n.tec, | support logic; impact depends on neighboring block and data flow.
| 4517 |                                     n.rec, | support logic; impact depends on neighboring block and data flow.
| 4518 |                                     match n.state { | branch dispatcher; arm changes alter behavior class hand
| 4519 |                                         BusState::ErrorActive => "EA", | support logic; impact depends on neigh
| 4520 |                                         BusState::ErrorPassive => "EP", | support logic; impact depends on nei
| 4521 |                                         BusState::BusOff => "BO", | support logic; impact depends on neighborin
```

```
| 4522 |                                    }, | support logic; impact depends on neighboring block and data flow. |
| 4523 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 4524 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 4525 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4526 |                                RichText::new(ecu.name) | support logic; impact depends on neighboring block ar
| 4527 |                                    .size(11.0) | support logic; impact depends on neighboring block and data f
| 4528 |                                    .color(Color32::from_gray(220)), | support logic; impact depends on neighbo
| 4529 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4530 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4531 |                                RichText::new(format!("0x{:02X}", ecu.sa)) | support logic; impact depends on n
| 4532 |                                    .size(11.0) | support logic; impact depends on neighboring block and data f
| 4533 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4534 |                                    .color(Color32::YELLOW), | support logic; impact depends on neighboring blo
| 4535 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4536 |                            ui.label(RichText::new(format!("{}", ecu.stage)).size(11.0).color(sc)); | support l
| 4537 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4538 |                                RichText::new( | support logic; impact depends on neighboring block and data fl
| 4539 |                                    ecu.online_at.map_or("—".into(), \|t\| format!("{:.2}s", t)), | support log
| 4540 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 4541 |                                .size(11.0) | support logic; impact depends on neighboring block and data flow
| 4542 |                                .color(Color32::from_gray(150)), | support logic; impact depends on neighboring
| 4543 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4544 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4545 |                                RichText::new(format!("{}", tec)) | support logic; impact depends on neighborin
| 4546 |                                    .size(11.0) | support logic; impact depends on neighboring block and data f
| 4547 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4548 |                                    .color(if tec > 0 { | support logic; impact depends on neighboring block ar
| 4549 |                                        Color32::YELLOW | support logic; impact depends on neighboring block ar
| 4550 |                                    } else { | support logic; impact depends on neighboring block and data flow
| 4551 |                                        Color32::GREEN | support logic; impact depends on neighboring block and
| 4552 |                                    }), | support logic; impact depends on neighboring block and data flow. |
| 4553 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4554 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4555 |                                RichText::new(format!("{}", rec)) | support logic; impact depends on neighborin
| 4556 |                                    .size(11.0) | support logic; impact depends on neighboring block and data f
| 4557 |                                    .monospace() | support logic; impact depends on neighboring block and data
| 4558 |                                    .color(if rec > 0 { | support logic; impact depends on neighboring block ar
| 4559 |                                        Color32::YELLOW | support logic; impact depends on neighboring block ar
| 4560 |                                    } else { | support logic; impact depends on neighboring block and data flow
| 4561 |                                        Color32::GREEN | support logic; impact depends on neighboring block and
| 4562 |                                    }), | support logic; impact depends on neighboring block and data flow. |
| 4563 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4564 |                            ui.label(RichText::new(bs).size(11.0).color(match bs { | support logic; impact depe
| 4565 |                                "BO" => Color32::RED, | support logic; impact depends on neighboring block and
| 4566 |                                "EP" => Color32::YELLOW, | support logic; impact depends on neighboring block a
| 4567 |                                _ => Color32::GREEN, | support logic; impact depends on neighboring block and d
| 4568 |                            })); | support logic; impact depends on neighboring block and data flow. |
| 4569 |                            ui.end_row(); | support logic; impact depends on neighboring block and data flow.
| 4570 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 4571 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 4572 |                ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4573 |                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4574 |                    RichText::new("CRITICAL NODES (TEC/REC or non-EA state):") | support logic; impact depends
| 4575 |                        .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 4576 |                        .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block a
| 4577 |                ); | support logic; impact depends on neighboring block and data flow. |
| 4578 |                for n in gw | iteration logic; may alter timing, throughput, and safety constraints. |
| 4579 |                    .nodes | support logic; impact depends on neighboring block and data flow. |
| 4580 |                    .iter() | support logic; impact depends on neighboring block and data flow. |
| 4581 |                    .filter(\|n\| n.tec > 0 \|\| n.rec > 0 \|\| !matches!(n.state, BusState::ErrorActive)) | su
| 4582 |                    .take(8) | support logic; impact depends on neighboring block and data flow. |
| 4583 |                { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4584 |                    ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4585 |                        RichText::new(format!( | support logic; impact depends on neighboring block and data fl
| 4586 |                            "SA 0x{:02X}  TEC:{} REC:{}  State:{}", | support logic; impact depends on neighbor
| 4587 |                            n.source_addr, n.tec, n.rec, n.state | support logic; impact depends on neighboring
| 4588 |                        )) | support logic; impact depends on neighboring block and data flow. |
| 4589 |                        .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 4590 |                        .monospace() | support logic; impact depends on neighboring block and data flow. |
| 4591 |                        .color(Color32::YELLOW), | support logic; impact depends on neighboring block and data
| 4592 |                    ); | support logic; impact depends on neighboring block and data flow. |
| 4593 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4594 |                ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 4595 |                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4596 |                    RichText::new("BUS ERROR LOG:") | support logic; impact depends on neighboring block and da
| 4597 |                        .size(11.0) | support logic; impact depends on neighboring block and data flow. |
```

```
| 4598 |                         .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block a
| 4599 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 4600 |                 for err in gw.error_log.iter().rev().take(40) { | iteration logic; may alter timing, throughpu
| 4601 |                     let sa = err | support logic; impact depends on neighboring block and data flow. |
| 4602 |                         .source_sa | support logic; impact depends on neighboring block and data flow. |
| 4603 |                         .map_or("----".to_string(), \|s\| format!("0x{:02X}", s)); | support logic; impact depe
| 4604 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4605 |                         RichText::new(format!( | support logic; impact depends on neighboring block and data f
| 4606 |                             "{:6.2}s SA:{}  {:?}  {}", | error propagation point; changes alter failure propaga
| 4607 |                             err.timestamp, sa, err.kind, err.description | support logic; impact depends on ne
| 4608 |                         )) | support logic; impact depends on neighboring block and data flow. |
| 4609 |                         .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 4610 |                         .monospace() | support logic; impact depends on neighboring block and data flow. |
| 4611 |                         .color(Color32::RED), | support logic; impact depends on neighboring block and data fl
| 4612 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 4613 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4614 |             }); | support logic; impact depends on neighboring block and data flow. |
| 4615 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4616 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4617 | (blank) | blank separator; no runtime behavior. |
| 4618 | // ═══════════════════════════════════════════════════════════════════════════════════ | comment
| 4619 | // TAB ENGINE (detailed ECM) | comment/documentation; changing affects understanding and intent traceability.
| 4620 | // ═══════════════════════════════════════════════════════════════════════════════════ | comment
| 4621 | impl App { | implementation binding; behavior semantics and trait conformance live here. |
| 4622 |     fn tab_engine(&self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flo
| 4623 |         let e = &self.bench.ecm; | support logic; impact depends on neighboring block and data flow. |
| 4624 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 4625 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 4626 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 4627 |                 ui.columns(3, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 4628 |                     // Col 0: Engine core | comment/documentation; changing affects understanding and intent t
| 4629 |                     egui::Frame::group(cols[0].style()) | support logic; impact depends on neighboring block a
| 4630 |                         .fill(Color32::from_gray(16)) | support logic; impact depends on neighboring block and
| 4631 |                         .show(&mut cols[0], \|ui\| { | support logic; impact depends on neighboring block and
| 4632 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4633 |                                 RichText::new("ENGINE CORE") | support logic; impact depends on neighboring bl
| 4634 |                                     .size(12.0) | support logic; impact depends on neighboring block and data
| 4635 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends o
| 4636 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4637 |                             let w = 140.0; | support logic; impact depends on neighboring block and data flow.
| 4638 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 4639 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4640 |                                 "Governor", | support logic; impact depends on neighboring block and data flow
| 4641 |                                 &format!("{}", e.governor_mode), | support logic; impact depends on neighborin
| 4642 |                                 Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 4643 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4644 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 4645 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4646 |                                 "Fuel Map", | support logic; impact depends on neighboring block and data flow
| 4647 |                                 &format!("{}", e.fuel_map), | support logic; impact depends on neighboring blo
| 4648 |                                 Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 4649 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4650 |                             bar_gauge(ui, "RPM", e.rpm, 3000.0, "rpm", w, Color32::GREEN); | support logic; im
| 4651 |                             bar_gauge(ui, "Load", e.percent_load, 150.0, "%", w, Color32::YELLOW); | support l
| 4652 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4653 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4654 |                                 "Torque", | support logic; impact depends on neighboring block and data flow.
| 4655 |                                 e.actual_torque_nm, | support logic; impact depends on neighboring block and d
| 4656 |                                 1200.0, | support logic; impact depends on neighboring block and data flow. |
| 4657 |                                 "Nm", | support logic; impact depends on neighboring block and data flow. |
| 4658 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4659 |                                 Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighborin
| 4660 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4661 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4662 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4663 |                                 "Power", | support logic; impact depends on neighboring block and data flow. |
| 4664 |                                 e.power_kw(), | support logic; impact depends on neighboring block and data fl
| 4665 |                                 260.0, | support logic; impact depends on neighboring block and data flow. |
| 4666 |                                 "kW", | support logic; impact depends on neighboring block and data flow. |
| 4667 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4668 |                                 Color32::from_rgb(100, 220, 100), | support logic; impact depends on neighbori
| 4669 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4670 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4671 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4672 |                                 "Throttle", | support logic; impact depends on neighboring block and data flow
| 4673 |                                 e.active_throttle, | support logic; impact depends on neighboring block and da
```

```
| 4674 |                                 100.0, | support logic; impact depends on neighboring block and data flow. |
| 4675 |                                 "%", | support logic; impact depends on neighboring block and data flow. |
| 4676 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4677 |                                 Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 4678 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4679 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 4680 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4681 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4682 |                                 "Rail P", | support logic; impact depends on neighboring block and data flow.
| 4683 |                                 e.fuel_rail_pressure_mpa * 10.0, | support logic; impact depends on neighborin
| 4684 |                                 2000.0, | support logic; impact depends on neighboring block and data flow. |
| 4685 |                                 "bar", | support logic; impact depends on neighboring block and data flow. |
| 4686 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4687 |                                 Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring
| 4688 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4689 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4690 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4691 |                                 "Fuel Prs", | support logic; impact depends on neighboring block and data flow
| 4692 |                                 e.fuel_pressure_kpa, | support logic; impact depends on neighboring block and
| 4693 |                                 600.0, | support logic; impact depends on neighboring block and data flow. |
| 4694 |                                 "kPa", | support logic; impact depends on neighboring block and data flow. |
| 4695 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4696 |                                 Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring
| 4697 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4698 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4699 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4700 |                                 "Fuel/h", | support logic; impact depends on neighboring block and data flow.
| 4701 |                                 e.fuel_rate_lph, | support logic; impact depends on neighboring block and data
| 4702 |                                 30.0, | support logic; impact depends on neighboring block and data flow. |
| 4703 |                                 "L/h", | support logic; impact depends on neighboring block and data flow. |
| 4704 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4705 |                                 Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 4706 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4707 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4708 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4709 |                                 "Boost", | support logic; impact depends on neighboring block and data flow. |
| 4710 |                                 e.boost_pressure_kpa, | support logic; impact depends on neighboring block and
| 4711 |                                 300.0, | support logic; impact depends on neighboring block and data flow. |
| 4712 |                                 "kPa", | support logic; impact depends on neighboring block and data flow. |
| 4713 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4714 |                                 Color32::from_rgb(180, 80, 255), | support logic; impact depends on neighborin
| 4715 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4716 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4717 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4718 |                                 "VGT", | support logic; impact depends on neighboring block and data flow. |
| 4719 |                                 e.vgt_position_pct, | support logic; impact depends on neighboring block and d
| 4720 |                                 100.0, | support logic; impact depends on neighboring block and data flow. |
| 4721 |                                 "%", | support logic; impact depends on neighboring block and data flow. |
| 4722 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4723 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block
| 4724 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4725 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4726 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4727 |                                 "EGR", | support logic; impact depends on neighboring block and data flow. |
| 4728 |                                 e.egr_valve_pct, | support logic; impact depends on neighboring block and data
| 4729 |                                 100.0, | support logic; impact depends on neighboring block and data flow. |
| 4730 |                                 "%", | support logic; impact depends on neighboring block and data flow. |
| 4731 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4732 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block
| 4733 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4734 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 4735 |                     // Col 1: Temperatures & pressures | comment/documentation; changing affects understanding
| 4736 |                     egui::Frame::group(cols[1].style()) | support logic; impact depends on neighboring block a
| 4737 |                         .fill(Color32::from_gray(16)) | support logic; impact depends on neighboring block and
| 4738 |                         .show(&mut cols[1], \|ui\| { | support logic; impact depends on neighboring block and
| 4739 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4740 |                                 RichText::new("THERMAL & PRESSURE") | support logic; impact depends on neighbo
| 4741 |                                     .size(12.0) | support logic; impact depends on neighboring block and data
| 4742 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends o
| 4743 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4744 |                             let w = 140.0; | support logic; impact depends on neighboring block and data flow.
| 4745 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4746 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4747 |                                 "Coolant", | support logic; impact depends on neighboring block and data flow.
| 4748 |                                 e.coolant_temp_c, | support logic; impact depends on neighboring block and data
| 4749 |                                 130.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 4750 |                                "°C", | support logic; impact depends on neighboring block and data flow. |
| 4751 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4752 |                                if e.coolant_temp_c > 100.0 { | runtime gate; condition changes can enable/disa
| 4753 |                                    Color32::RED | support logic; impact depends on neighboring block and data
| 4754 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 4755 |                                    Color32::GREEN | support logic; impact depends on neighboring block and da
| 4756 |                                }, | support logic; impact depends on neighboring block and data flow. |
| 4757 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4758 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4759 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4760 |                                "Oil", | support logic; impact depends on neighboring block and data flow. |
| 4761 |                                e.oil_temp_c, | support logic; impact depends on neighboring block and data flo
| 4762 |                                160.0, | support logic; impact depends on neighboring block and data flow. |
| 4763 |                                "°C", | support logic; impact depends on neighboring block and data flow. |
| 4764 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4765 |                                Color32::from_rgb(200, 150, 50), | support logic; impact depends on neighboring
| 4766 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4767 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4768 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4769 |                                "Exhaust", | support logic; impact depends on neighboring block and data flow.
| 4770 |                                e.exhaust_temp_c, | support logic; impact depends on neighboring block and data
| 4771 |                                900.0, | support logic; impact depends on neighboring block and data flow. |
| 4772 |                                "°C", | support logic; impact depends on neighboring block and data flow. |
| 4773 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4774 |                                Color32::from_rgb(255, 120, 30), | support logic; impact depends on neighboring
| 4775 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4776 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4777 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4778 |                                "Intake", | support logic; impact depends on neighboring block and data flow.
| 4779 |                                e.intake_temp_c, | support logic; impact depends on neighboring block and data
| 4780 |                                80.0, | support logic; impact depends on neighboring block and data flow. |
| 4781 |                                "°C", | support logic; impact depends on neighboring block and data flow. |
| 4782 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4783 |                                Color32::from_gray(200), | support logic; impact depends on neighboring block
| 4784 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4785 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4786 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4787 |                                "Fuel T", | support logic; impact depends on neighboring block and data flow.
| 4788 |                                e.fuel_temp_c, | support logic; impact depends on neighboring block and data f
| 4789 |                                80.0, | support logic; impact depends on neighboring block and data flow. |
| 4790 |                                "°C", | support logic; impact depends on neighboring block and data flow. |
| 4791 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4792 |                                Color32::from_gray(200), | support logic; impact depends on neighboring block
| 4793 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4794 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow
| 4795 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4796 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4797 |                                "Oil Prs", | support logic; impact depends on neighboring block and data flow.
| 4798 |                                e.oil_pressure_kpa, | support logic; impact depends on neighboring block and da
| 4799 |                                700.0, | support logic; impact depends on neighboring block and data flow. |
| 4800 |                                "kPa", | support logic; impact depends on neighboring block and data flow. |
| 4801 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4802 |                                if e.oil_pressure_kpa < 100.0 { | runtime gate; condition changes can enable/di
| 4803 |                                    Color32::RED | support logic; impact depends on neighboring block and data
| 4804 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 4805 |                                    Color32::GREEN | support logic; impact depends on neighboring block and da
| 4806 |                                }, | support logic; impact depends on neighboring block and data flow. |
| 4807 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4808 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4809 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4810 |                                "Coolant P", | support logic; impact depends on neighboring block and data flow
| 4811 |                                e.coolant_pres_kpa, | support logic; impact depends on neighboring block and da
| 4812 |                                150.0, | support logic; impact depends on neighboring block and data flow. |
| 4813 |                                "kPa", | support logic; impact depends on neighboring block and data flow. |
| 4814 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4815 |                                Color32::from_gray(200), | support logic; impact depends on neighboring block
| 4816 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4817 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4818 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4819 |                                "Air Filter", | support logic; impact depends on neighboring block and data flo
| 4820 |                                e.air_filter_dp_kpa, | support logic; impact depends on neighboring block and
| 4821 |                                10.0, | support logic; impact depends on neighboring block and data flow. |
| 4822 |                                "kPa", | support logic; impact depends on neighboring block and data flow. |
| 4823 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4824 |                                if e.air_filter_dp_kpa > 5.0 { | runtime gate; condition changes can enable/dis
| 4825 |                                    Color32::YELLOW | support logic; impact depends on neighboring block and da
```

```
| 4826 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 4827 |                                     Color32::GREEN | support logic; impact depends on neighboring block and dat
| 4828 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 4829 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4830 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 4831 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 4832 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4833 |                                 "Alt V", | support logic; impact depends on neighboring block and data flow. |
| 4834 |                                 &format!("{:.1}V", e.alternator_v), | support logic; impact depends on neighbor
| 4835 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block a
| 4836 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4837 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 4838 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4839 |                                 "Batt V", | support logic; impact depends on neighboring block and data flow.
| 4840 |                                 &format!("{:.1}V", self.bench.bcm.battery_voltage), | support logic; impact dep
| 4841 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block a
| 4842 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4843 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 4844 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4845 |                                 "Engine H", | support logic; impact depends on neighboring block and data flow
| 4846 |                                 &format!("{:.1}h", e.engine_hours), | support logic; impact depends on neighbor
| 4847 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block a
| 4848 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4849 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 4850 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4851 |                                 "Svc due", | support logic; impact depends on neighboring block and data flow.
| 4852 |                                 &format!("{:.1}h", e.service_hours_left), | support logic; impact depends on ne
| 4853 |                                 if e.service_hours_left < 50.0 { | runtime gate; condition changes can enable/d
| 4854 |                                     Color32::RED | support logic; impact depends on neighboring block and data
| 4855 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 4856 |                                     Color32::GREEN | support logic; impact depends on neighboring block and dat
| 4857 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 4858 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4859 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 4860 |                     // Col 2: Aftertreatment + DTCs | comment/documentation; changing affects understanding and
| 4861 |                     egui::Frame::group(cols[2].style()) | support logic; impact depends on neighboring block an
| 4862 |                         .fill(Color32::from_gray(16)) | support logic; impact depends on neighboring block and
| 4863 |                         .show(&mut cols[2], \|ui\| { | support logic; impact depends on neighboring block and d
| 4864 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4865 |                                 RichText::new("AFTERTREATMENT & DTCS") | support logic; impact depends on neigh
| 4866 |                                     .size(12.0) | support logic; impact depends on neighboring block and data f
| 4867 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 4868 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4869 |                             let w = 130.0; | support logic; impact depends on neighboring block and data flow.
| 4870 |                             let ac = match e.aftertreatment { | support logic; impact depends on neighboring bl
| 4871 |                                 AftertreatmentState::Normal => Color32::GREEN, | support logic; impact depends
| 4872 |                                 AftertreatmentState::DpfRegen => Color32::from_rgb(200, 80, 200), | support log
| 4873 |                                 AftertreatmentState::ScrDegraded => Color32::YELLOW, | support logic; impact de
| 4874 |                                 _ => Color32::RED, | support logic; impact depends on neighboring block and dat
| 4875 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 4876 |                             digital_readout(ui, "State", &format!("{}", e.aftertreatment), ac); | support logic
| 4877 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4878 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4879 |                                 "DPF Soot", | support logic; impact depends on neighboring block and data flow
| 4880 |                                 e.dpf_soot_pct, | support logic; impact depends on neighboring block and data f
| 4881 |                                 100.0, | support logic; impact depends on neighboring block and data flow. |
| 4882 |                                 "%", | support logic; impact depends on neighboring block and data flow. |
| 4883 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4884 |                                 if e.dpf_soot_pct > 75.0 { | runtime gate; condition changes can enable/disable
| 4885 |                                     Color32::RED | support logic; impact depends on neighboring block and data
| 4886 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 4887 |                                     Color32::from_rgb(180, 100, 220) | support logic; impact depends on neighbo
| 4888 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 4889 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4890 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4891 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4892 |                                 "DPF Temp", | support logic; impact depends on neighboring block and data flow
| 4893 |                                 e.dpf_temp_c, | support logic; impact depends on neighboring block and data flo
| 4894 |                                 700.0, | support logic; impact depends on neighboring block and data flow. |
| 4895 |                                 "°C", | support logic; impact depends on neighboring block and data flow. |
| 4896 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 4897 |                                 Color32::from_rgb(255, 130, 30), | support logic; impact depends on neighboring
| 4898 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 4899 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4900 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 4901 |                                 "DEF Level", | support logic; impact depends on neighboring block and data flow
```

```
| 4902 |                                e.def_level_pct, | support logic; impact depends on neighboring block and data
| 4903 |                                100.0, | support logic; impact depends on neighboring block and data flow. |
| 4904 |                                "%", | support logic; impact depends on neighboring block and data flow. |
| 4905 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4906 |                                if e.def_level_pct < 10.0 { | runtime gate; condition changes can enable/disab
| 4907 |                                    Color32::RED | support logic; impact depends on neighboring block and data
| 4908 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 4909 |                                    Color32::from_rgb(0, 210, 210) | support logic; impact depends on neighbor
| 4910 |                                }, | support logic; impact depends on neighboring block and data flow. |
| 4911 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4912 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 4913 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4914 |                                "SCR Eff", | support logic; impact depends on neighboring block and data flow.
| 4915 |                                e.scr_efficiency_pct, | support logic; impact depends on neighboring block and
| 4916 |                                100.0, | support logic; impact depends on neighboring block and data flow. |
| 4917 |                                "%", | support logic; impact depends on neighboring block and data flow. |
| 4918 |                                w, | support logic; impact depends on neighboring block and data flow. |
| 4919 |                                if e.scr_efficiency_pct < 80.0 { | runtime gate; condition changes can enable/
| 4920 |                                    Color32::YELLOW | support logic; impact depends on neighboring block and d
| 4921 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 4922 |                                    Color32::GREEN | support logic; impact depends on neighboring block and da
| 4923 |                                }, | support logic; impact depends on neighboring block and data flow. |
| 4924 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4925 |                            digital_readout( | support logic; impact depends on neighboring block and data flo
| 4926 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4927 |                                "NOx raw", | support logic; impact depends on neighboring block and data flow.
| 4928 |                                &format!("{:.0} ppm", e.nox_raw_ppm), | support logic; impact depends on neigh
| 4929 |                                Color32::from_gray(180), | support logic; impact depends on neighboring block
| 4930 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4931 |                            digital_readout( | support logic; impact depends on neighboring block and data flo
| 4932 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 4933 |                                "NOx tail", | support logic; impact depends on neighboring block and data flow
| 4934 |                                &format!("{:.0} ppm", e.nox_tailpipe_ppm), | support logic; impact depends on
| 4935 |                                if e.nox_tailpipe_ppm > 500.0 { | runtime gate; condition changes can enable/d
| 4936 |                                    Color32::RED | support logic; impact depends on neighboring block and data
| 4937 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 4938 |                                    Color32::from_gray(180) | support logic; impact depends on neighboring bloc
| 4939 |                                }, | support logic; impact depends on neighboring block and data flow. |
| 4940 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4941 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow
| 4942 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 4943 |                                RichText::new("ACTIVE DTCs:") | support logic; impact depends on neighboring b
| 4944 |                                    .size(11.0) | support logic; impact depends on neighboring block and data
| 4945 |                                    .color(Color32::from_gray(140)), | support logic; impact depends on neighb
| 4946 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 4947 |                            if e.active_dtcs.is_empty() { | runtime gate; condition changes can enable/disable
| 4948 |                                ui.label(RichText::new("✓ None").size(11.0).color(Color32::GREEN)); | support
| 4949 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 4950 |                            for dtc in &e.active_dtcs { | iteration logic; may alter timing, throughput, and s
| 4951 |                                let c = match dtc.severity { | support logic; impact depends on neighboring bl
| 4952 |                                    DtcSeverity::Red => Color32::RED, | support logic; impact depends on neigh
| 4953 |                                    DtcSeverity::Amber => Color32::GOLD, | support logic; impact depends on ne
| 4954 |                                    _ => Color32::LIGHT_BLUE, | support logic; impact depends on neighboring b
| 4955 |                                }; | scope delimiter; structural only, but wrong placement changes ownership/l
| 4956 |                                ui.label( | support logic; impact depends on neighboring block and data flow.
| 4957 |                                    RichText::new(format!( | support logic; impact depends on neighboring bloc
| 4958 |                                        "SPN:{} FMI:{} ×{}  {}", | support logic; impact depends on neighborin
| 4959 |                                        dtc.spn, dtc.fmi, dtc.count, dtc.desc | support logic; impact depends
| 4960 |                                    )) | support logic; impact depends on neighboring block and data flow. |
| 4961 |                                    .size(10.5) | support logic; impact depends on neighboring block and data
| 4962 |                                    .color(c), | support logic; impact depends on neighboring block and data f
| 4963 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 4964 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 4965 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 4966 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 4967 |                }); | support logic; impact depends on neighboring block and data flow. |
| 4968 |        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4969 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4970 | (blank) | blank separator; no runtime behavior. |
| 4971 | // ━━━
```

```
| 4978 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4979 | (blank) | blank separator; no runtime behavior. |
| 4980 |     fn dtc_dm1_payload_hex(amber: bool, red: bool, mil: bool, spn: u32, fmi: u8) -> String { | function signatu
| 4981 |         let dm1 = auto_breaking::j1939::Builder::dm1( | support logic; impact depends on neighboring block and
| 4982 |             0.0, | support logic; impact depends on neighboring block and data flow. |
| 4983 |             amber, | support logic; impact depends on neighboring block and data flow. |
| 4984 |             red, | support logic; impact depends on neighboring block and data flow. |
| 4985 |             mil, | support logic; impact depends on neighboring block and data flow. |
| 4986 |             spn, | support logic; impact depends on neighboring block and data flow. |
| 4987 |             fmi, | support logic; impact depends on neighboring block and data flow. |
| 4988 |             auto_breaking::j1939::addr::ECM_1, | support logic; impact depends on neighboring block and data f
| 4989 |         ); | support logic; impact depends on neighboring block and data flow. |
| 4990 |         dm1.data | support logic; impact depends on neighboring block and data flow. |
| 4991 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 4992 |             .map(\|b\| format!("{:02X}", b)) | support logic; impact depends on neighboring block and data flo
| 4993 |             .collect::<Vec<_>>() | support logic; impact depends on neighboring block and data flow. |
| 4994 |             .join(" ") | support logic; impact depends on neighboring block and data flow. |
| 4995 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 4996 | (blank) | blank separator; no runtime behavior. |
| 4997 |     fn fault_selection_hint(selected: auto_breaking::FaultType) -> Option<(u32, u8, &'static str)> { | functio
| 4998 |         match selected { | branch dispatcher; arm changes alter behavior class handling. |
| 4999 |             auto_breaking::FaultType::HighCoolantTemp => { | support logic; impact depends on neighboring bloc
| 5000 |                 Some((110, 0, "Coolant temp above severe threshold")) | support logic; impact depends on neigh
| 5001 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5002 |             auto_breaking::FaultType::LowOilPressure => { | support logic; impact depends on neighboring block
| 5003 |                 Some((100, 1, "Oil pressure below severe threshold")) | support logic; impact depends on neigh
| 5004 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5005 |             auto_breaking::FaultType::LowFuelPressure => { | support logic; impact depends on neighboring bloc
| 5006 |                 Some((94, 1, "Fuel delivery pressure below threshold")) | support logic; impact depends on nei
| 5007 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5008 |             auto_breaking::FaultType::CriticalDefLevel => { | support logic; impact depends on neighboring bloc
| 5009 |                 Some((3361, 1, "DEF critically low (derate trigger)")) | support logic; impact depends on neig
| 5010 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5011 |             auto_breaking::FaultType::HighDpfSoot => { | support logic; impact depends on neighboring block an
| 5012 |                 Some((3251, 16, "DPF soot above high threshold")) | support logic; impact depends on neighbori
| 5013 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5014 |             _ => None, | support logic; impact depends on neighboring block and data flow. |
| 5015 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5016 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5017 | (blank) | blank separator; no runtime behavior. |
| 5018 |     fn tab_faults(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control
| 5019 |         let dtc_now = self.bench.ecm.active_dtcs.len(); | support logic; impact depends on neighboring block a
| 5020 |         let dtc_delta = dtc_now as isize - self.fault_last_dtc_count as isize; | support logic; impact depends
| 5021 |         if dtc_delta != 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 5022 |             self.fault_feedback = format!( | support logic; impact depends on neighboring block and data flow.
| 5023 |                 "ECM DTC update observed: {} -> {} ({:+})", | support logic; impact depends on neighboring bloc
| 5024 |                 self.fault_last_dtc_count, dtc_now, dtc_delta | support logic; impact depends on neighboring b
| 5025 |             ); | support logic; impact depends on neighboring block and data flow. |
| 5026 |             self.fault_last_dtc_count = dtc_now; | support logic; impact depends on neighboring block and data
| 5027 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5028 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 5029 |             .id_salt("faults::page_scroll") | support logic; impact depends on neighboring block and data flow
| 5030 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 5031 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 5032 |         ui.columns(2, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 5033 |             // Left: active DTCs | comment/documentation; changing affects understanding and intent traceabili
| 5034 |             let ui = &mut cols[0]; | support logic; impact depends on neighboring block and data flow. |
| 5035 |             egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data flow.
| 5036 |                 .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data fl
| 5037 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 5038 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5039 |                         RichText::new("ACTIVE DTCs (DM1)") | support logic; impact depends on neighboring bloc
| 5040 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 5041 |                             .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighb
| 5042 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5043 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5044 |                     let dtcs_empty = self.bench.ecm.active_dtcs.is_empty(); | support logic; impact depends on
| 5045 |                     if dtcs_empty { | runtime gate; condition changes can enable/disable critical paths. |
| 5046 |                         ui.add_space(10.0); | support logic; impact depends on neighboring block and data flow
| 5047 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5048 |                             RichText::new("✓  No active fault codes") | support logic; impact depends on neigh
| 5049 |                                 .size(13.0) | support logic; impact depends on neighboring block and data flow
| 5050 |                                 .color(Color32::GREEN), | support logic; impact depends on neighboring block a
| 5051 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5052 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 5053 |                         ScrollArea::vertical() | support logic; impact depends on neighboring block and data f
```

```
| 5054 |                                .id_salt("faults::active_dtcs_scroll") | support logic; impact depends on neighbori
| 5055 |                                .max_height(250.0) | support logic; impact depends on neighboring block and data fl
| 5056 |                                .auto_shrink([false, false]) | support logic; impact depends on neighboring block a
| 5057 |                                .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data fl
| 5058 |                                    for dtc in &self.bench.ecm.active_dtcs { | iteration logic; may alter timing, i
| 5059 |                                        let c = match dtc.severity { | support logic; impact depends on neighboring
| 5060 |                                            DtcSeverity::Red => Color32::RED, | support logic; impact depends on ne
| 5061 |                                            DtcSeverity::Amber => Color32::GOLD, | support logic; impact depends on
| 5062 |                                            DtcSeverity::Mil => Color32::LIGHT_BLUE, | support logic; impact depend
| 5063 |                                            DtcSeverity::Protect => Color32::from_rgb(0, 210, 210), | support logic
| 5064 |                                        }; | scope delimiter; structural only, but wrong placement changes ownership
| 5065 |                                        egui::Frame::group(ui.style()) | support logic; impact depends on neighbori
| 5066 |                                            .fill(Color32::from_gray(20)) | support logic; impact depends on neighb
| 5067 |                                            .show(ui, \|ui\| { | support logic; impact depends on neighboring block
| 5068 |                                                ui.horizontal(\|ui\| { | support logic; impact depends on neighbori
| 5069 |                                                    ui.label( | support logic; impact depends on neighboring block
| 5070 |                                                        RichText::new(format!("SPN {:6}", dtc.spn)) | support logic
| 5071 |                                                            .size(13.0) | support logic; impact depends on neighbor
| 5072 |                                                            .color(c) | support logic; impact depends on neighborin
| 5073 |                                                            .strong(), | support logic; impact depends on neighbori
| 5074 |                                                    ); | support logic; impact depends on neighboring block and dat
| 5075 |                                                    ui.label( | support logic; impact depends on neighboring block
| 5076 |                                                        RichText::new(format!("FMI {:2}", dtc.fmi)) | support logic
| 5077 |                                                            .size(12.0) | support logic; impact depends on neighbor
| 5078 |                                                            .color(Color32::GOLD), | support logic; impact depends
| 5079 |                                                    ); | support logic; impact depends on neighboring block and dat
| 5080 |                                                    ui.label( | support logic; impact depends on neighboring block
| 5081 |                                                        RichText::new(format!("×{}", dtc.count)) | support logic; i
| 5082 |                                                            .size(11.0) | support logic; impact depends on neighbor
| 5083 |                                                            .color(Color32::GRAY), | support logic; impact depends
| 5084 |                                                    ); | support logic; impact depends on neighboring block and dat
| 5085 |                                                    ui.label( | support logic; impact depends on neighboring block
| 5086 |                                                        RichText::new(format!("[{}]", dtc.severity)) | support logi
| 5087 |                                                            .size(11.0) | support logic; impact depends on neighbor
| 5088 |                                                            .color(c), | support logic; impact depends on neighbori
| 5089 |                                                    ); | support logic; impact depends on neighboring block and dat
| 5090 |                                                }); | support logic; impact depends on neighboring block and data f
| 5091 |                                                ui.label( | support logic; impact depends on neighboring block and
| 5092 |                                                    RichText::new(dtc.desc) | support logic; impact depends on neig
| 5093 |                                                        .size(11.0) | support logic; impact depends on neighboring
| 5094 |                                                        .color(Color32::from_gray(200)), | support logic; impact de
| 5095 |                                                ); | support logic; impact depends on neighboring block and data fl
| 5096 |                                                let mem_addr = Self::dtc_fault_storage_addr(dtc.spn, dtc.fmi); | su
| 5097 |                                                let dm1_hex = Self::dtc_dm1_payload_hex( | support logic; impact de
| 5098 |                                                    matches!(dtc.severity, DtcSeverity::Amber), | support logic; im
| 5099 |                                                    matches!(dtc.severity, DtcSeverity::Red), | support logic; impa
| 5100 |                                                    matches!(dtc.severity, DtcSeverity::Mil), | support logic; impa
| 5101 |                                                    dtc.spn, | support logic; impact depends on neighboring block a
| 5102 |                                                    dtc.fmi, | support logic; impact depends on neighboring block a
| 5103 |                                                ); | support logic; impact depends on neighboring block and data fl
| 5104 |                                                ui.label( | support logic; impact depends on neighboring block and
| 5105 |                                                    RichText::new(format!( | support logic; impact depends on neigh
| 5106 |                                                        "ECM fault storage addr: 0x{mem_addr:08X} (NVM.FaultStorage
| 5107 |                                                    )) | support logic; impact depends on neighboring block and dat
| 5108 |                                                    .size(9.7) | support logic; impact depends on neighboring block
| 5109 |                                                    .monospace() | support logic; impact depends on neighboring bl
| 5110 |                                                    .color(Color32::from_rgb(140, 210, 255)), | support logic; impa
| 5111 |                                                ); | support logic; impact depends on neighboring block and data fl
| 5112 |                                                ui.label( | support logic; impact depends on neighboring block and
| 5113 |                                                    RichText::new( | support logic; impact depends on neighboring b
| 5114 |                                                        "Diagnostic request -> ECM SA 0x00: UDS 19 02 FF (ReadDTC b
| 5115 |                                                            .to_string(), | support logic; impact depends on neighb
| 5116 |                                                    ) | support logic; impact depends on neighboring block and data
| 5117 |                                                    .size(9.6) | support logic; impact depends on neighboring block
| 5118 |                                                    .monospace() | support logic; impact depends on neighboring bl
| 5119 |                                                    .color(Color32::from_gray(170)), | support logic; impact depend
| 5120 |                                                ); | support logic; impact depends on neighboring block and data fl
| 5121 |                                                ui.label( | support logic; impact depends on neighboring block and
| 5122 |                                                    RichText::new(format!( | support logic; impact depends on neigh
| 5123 |                                                        "ECM recognition broadcast -> J1939 DM1 PGN 65226 DATA [{}]
| 5124 |                                                        dm1_hex | support logic; impact depends on neighboring bloc
| 5125 |                                                    )) | support logic; impact depends on neighboring block and dat
| 5126 |                                                    .size(9.6) | support logic; impact depends on neighboring block
| 5127 |                                                    .monospace() | support logic; impact depends on neighboring bl
| 5128 |                                                    .color(Color32::from_gray(170)), | support logic; impact depend
| 5129 |                                                ); | support logic; impact depends on neighboring block and data fl
```

```
| 5130 |                                        }); | support logic; impact depends on neighboring block and data flow
| 5131 |                                } | scope delimiter; structural only, but wrong placement changes ownership/li
| 5132 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 5133 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5134 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5135 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5136 |                            RichText::new(format!( | support logic; impact depends on neighboring block and data f
| 5137 |                                "Fault pipeline: selected -> inject -> ECM evaluate -> DM1 broadcast \| DTC now: {
| 5138 |                                dtc_now, dtc_delta | support logic; impact depends on neighboring block and data f
| 5139 |                            )) | support logic; impact depends on neighboring block and data flow. |
| 5140 |                            .size(10.0) | support logic; impact depends on neighboring block and data flow. |
| 5141 |                            .color(Color32::from_gray(150)), | support logic; impact depends on neighboring block a
| 5142 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5143 |                        if !self.fault_feedback.is_empty() { | runtime gate; condition changes can enable/disable c
| 5144 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5145 |                                RichText::new(format!("Last action: {}", self.fault_feedback)) | support logic; imp
| 5146 |                                    .size(10.2) | support logic; impact depends on neighboring block and data flow
| 5147 |                                    .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighboring blo
| 5148 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 5149 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5150 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5151 |                        // Warning lamps | comment/documentation; changing affects understanding and intent traceal
| 5152 |                        ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 5153 |                            let e = &self.bench.ecm; | support logic; impact depends on neighboring block and data
| 5154 |                            for (lbl, on, c) in [ | iteration logic; may alter timing, throughput, and safety cons
| 5155 |                                ("■ RED", e.red_lamp, Color32::RED), | support logic; impact depends on neighborin
| 5156 |                                ("■ AMB", e.amber_lamp, Color32::GOLD), | support logic; impact depends on neighbo
| 5157 |                                ("■ MIL", e.mil_active, Color32::LIGHT_BLUE), | support logic; impact depends on n
| 5158 |                                ("■ PROT", e.protect_lamp, Color32::from_rgb(0, 210, 210)), | support logic; impac
| 5159 |                            ] { | support logic; impact depends on neighboring block and data flow. |
| 5160 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5161 |                                    RichText::new(lbl) | support logic; impact depends on neighboring block and da
| 5162 |                                        .size(12.0) | support logic; impact depends on neighboring block and data
| 5163 |                                        .color(if on { c } else { Color32::from_gray(55) }) | support logic; impac
| 5164 |                                        .strong(), | support logic; impact depends on neighboring block and data f
| 5165 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 5166 |                                ui.add_space(4.0); | support logic; impact depends on neighboring block and data f
| 5167 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5168 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 5169 |                        ui.add_space(6.0); | support logic; impact depends on neighboring block and data flow. |
| 5170 |                        if ui | runtime gate; condition changes can enable/disable critical paths. |
| 5171 |                            .add( | support logic; impact depends on neighboring block and data flow. |
| 5172 |                                Button::new(RichText::new("■ CLEAR ALL DTCs").size(12.0)) | support logic; impact
| 5173 |                                    .fill(Color32::from_rgb(50, 22, 22)), | support logic; impact depends on neigh
| 5174 |                            ) | support logic; impact depends on neighboring block and data flow. |
| 5175 |                            .clicked() | support logic; impact depends on neighboring block and data flow. |
| 5176 |                        { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5177 |                            self.cmds.push(Cmd::ClearDtcs); | support logic; impact depends on neighboring block a
| 5178 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5179 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 5180 | (blank) | blank separator; no runtime behavior. |
| 5181 |                // Right: fault injection | comment/documentation; changing affects understanding and intent tracea
| 5182 |                let ui = &mut cols[1]; | support logic; impact depends on neighboring block and data flow. |
| 5183 |                egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data flow.
| 5184 |                    .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data fl
| 5185 |                    .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 5186 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5187 |                            RichText::new("FAULT INJECTION PANEL") | support logic; impact depends on neighboring b
| 5188 |                                .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 5189 |                                .color(Color32::from_rgb(200, 80, 80)), | support logic; impact depends on neighbo
| 5190 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5191 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5192 |                        ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow.
| 5193 |                            .id_salt("faults::selector_scroll") | support logic; impact depends on neighboring bloc
| 5194 |                            .max_height(340.0) | support logic; impact depends on neighboring block and data flow.
| 5195 |                            .auto_shrink([false, false]) | support logic; impact depends on neighboring block and
| 5196 |                            .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 5197 |                                for (i, ft) in FAULT_TYPES.iter().enumerate() { | iteration logic; may alter timing
| 5198 |                                    ui.push_id(format!("fault::selector::{}", i), \|ui\| { | support logic; impact
| 5199 |                                        let sel = i == self.fault_idx; | support logic; impact depends on neighbor
| 5200 |                                        let act = | support logic; impact depends on neighboring block and data flo
| 5201 |                                            self.bench.fault_active && self.bench.selected_fault == *ft; | support
| 5202 |                                        let fill = if act { | support logic; impact depends on neighboring block a
| 5203 |                                            Color32::from_rgb(70, 15, 15) | support logic; impact depends on neigh
| 5204 |                                        } else if sel { | support logic; impact depends on neighboring block and da
| 5205 |                                            Color32::from_rgb(25, 45, 75) | support logic; impact depends on neigh
```

```
| 5206 |                                     } else { | support logic; impact depends on neighboring block and data flo
| 5207 |                                         Color32::TRANSPARENT | support logic; impact depends on neighboring bl
| 5208 |                                     }; | scope delimiter; structural only, but wrong placement changes ownersh
| 5209 |                                     let col = if act { | support logic; impact depends on neighboring block an
| 5210 |                                         Color32::RED | support logic; impact depends on neighboring block and
| 5211 |                                     } else if sel { | support logic; impact depends on neighboring block and d
| 5212 |                                         Color32::from_rgb(180, 220, 255) | support logic; impact depends on ne
| 5213 |                                     } else { | support logic; impact depends on neighboring block and data flo
| 5214 |                                         Color32::from_gray(155) | support logic; impact depends on neighboring
| 5215 |                                     }; | scope delimiter; structural only, but wrong placement changes ownersh
| 5216 |                                     let btn = Button::new( | support logic; impact depends on neighboring bloc
| 5217 |                                         RichText::new(format!("{}", ft)).size(11.0).color(col), | support logi
| 5218 |                                     ) | support logic; impact depends on neighboring block and data flow. |
| 5219 |                                     .fill(fill) | support logic; impact depends on neighboring block and data
| 5220 |                                     .min_size(Vec2::new(ui.available_width(), 22.0)); | support logic; impact
| 5221 |                                     if ui.add(btn).clicked() { | runtime gate; condition changes can enable/di
| 5222 |                                         self.cmds.push(Cmd::SelectFault(i)); | support logic; impact depends o
| 5223 |                                     } | scope delimiter; structural only, but wrong placement changes ownershi
| 5224 |                                 }); | support logic; impact depends on neighboring block and data flow. |
| 5225 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 5226 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 5227 |                     ui.add_space(6.0); | support logic; impact depends on neighboring block and data flow. |
| 5228 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 5229 |                         let ic = if self.bench.fault_active { | support logic; impact depends on neighboring b
| 5230 |                             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 5231 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5232 |                             Color32::from_rgb(160, 35, 35) | support logic; impact depends on neighboring bloc
| 5233 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 5234 |                         let lbl = if self.bench.fault_active { | support logic; impact depends on neighboring
| 5235 |                             "■ ACTIVE — Inject Again" | support logic; impact depends on neighboring block and
| 5236 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5237 |                             "■ INJECT FAULT" | support logic; impact depends on neighboring block and data flo
| 5238 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 5239 |                         if ui | runtime gate; condition changes can enable/disable critical paths. |
| 5240 |                             .add( | support logic; impact depends on neighboring block and data flow. |
| 5241 |                                 Button::new(RichText::new(lbl).size(13.0)) | support logic; impact depends on
| 5242 |                                     .fill(ic) | support logic; impact depends on neighboring block and data fl
| 5243 |                                     .min_size(Vec2::new(185.0, 36.0)), | support logic; impact depends on neig
| 5244 |                             ) | support logic; impact depends on neighboring block and data flow. |
| 5245 |                             .clicked() | support logic; impact depends on neighboring block and data flow. |
| 5246 |                         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5247 |                             self.cmds.push(Cmd::InjectFault); | support logic; impact depends on neighboring b
| 5248 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5249 |                         if ui | runtime gate; condition changes can enable/disable critical paths. |
| 5250 |                             .add( | support logic; impact depends on neighboring block and data flow. |
| 5251 |                                 Button::new(RichText::new("■ CLEAR").size(13.0)) | support logic; impact depen
| 5252 |                                     .fill(Color32::from_rgb(22, 68, 32)) | support logic; impact depends on ne
| 5253 |                                     .min_size(Vec2::new(90.0, 36.0)), | support logic; impact depends on neigh
| 5254 |                             ) | support logic; impact depends on neighboring block and data flow. |
| 5255 |                             .clicked() | support logic; impact depends on neighboring block and data flow. |
| 5256 |                         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5257 |                             self.cmds.push(Cmd::ClearFaults); | support logic; impact depends on neighboring b
| 5258 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5259 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 5260 |                     if self.bench.fault_active { | runtime gate; condition changes can enable/disable critical
| 5261 |                         ui.add_space(4.0); | support logic; impact depends on neighboring block and data flow.
| 5262 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5263 |                             RichText::new(format!("■ ACTIVE FAULT: {}", self.bench.selected_fault)) | support
| 5264 |                                 .size(11.5) | support logic; impact depends on neighboring block and data flow
| 5265 |                                 .color(Color32::RED) | support logic; impact depends on neighboring block and
| 5266 |                                 .strong(), | support logic; impact depends on neighboring block and data flow.
| 5267 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5268 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5269 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5270 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5271 |                         RichText::new("ECM DTC RECOGNITION GUIDE") | support logic; impact depends on neighbor
| 5272 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5273 |                             .color(Color32::from_rgb(110, 180, 255)) | support logic; impact depends on neighb
| 5274 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 5275 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5276 |                     if let Some((spn, fmi, msg)) = | runtime gate; condition changes can enable/disable critica
| 5277 |                         Self::fault_selection_hint(self.bench.selected_fault) | support logic; impact depends
| 5278 |                     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5279 |                         let addr = Self::dtc_fault_storage_addr(spn, fmi); | support logic; impact depends on
| 5280 |                         let dm1_hex = Self::dtc_dm1_payload_hex(true, false, false, spn, fmi); | support logic
| 5281 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
```

```
| 5282 |                            RichText::new(format!( | support logic; impact depends on neighboring block and dat
| 5283 |                                "Selected fault maps to SPN {spn} FMI {fmi} -> addr 0x{addr:08X}" | support log
| 5284 |                            )) | support logic; impact depends on neighboring block and data flow. |
| 5285 |                            .size(10.2) | support logic; impact depends on neighboring block and data flow. |
| 5286 |                            .monospace() | support logic; impact depends on neighboring block and data flow. |
| 5287 |                            .color(Color32::from_rgb(140, 210, 255)), | support logic; impact depends on neighb
| 5288 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5289 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5290 |                            RichText::new(format!("Trigger condition: {msg}")) | support logic; impact depends
| 5291 |                                .size(10.0) | support logic; impact depends on neighboring block and data flow
| 5292 |                                .color(Color32::from_gray(180)), | support logic; impact depends on neighboring
| 5293 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5294 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5295 |                            RichText::new("Message to query/ack in diagnostics: UDS 19 02 FF -> ECM SA 0x00")
| 5296 |                                .size(9.8) | support logic; impact depends on neighboring block and data flow.
| 5297 |                                .monospace() | support logic; impact depends on neighboring block and data flow
| 5298 |                                .color(Color32::from_gray(170)), | support logic; impact depends on neighboring
| 5299 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5300 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5301 |                            RichText::new(format!( | support logic; impact depends on neighboring block and dat
| 5302 |                                "Expected ECM DM1 payload for this DTC: [{}]", | support logic; impact depends
| 5303 |                                dm1_hex | support logic; impact depends on neighboring block and data flow. |
| 5304 |                            )) | support logic; impact depends on neighboring block and data flow. |
| 5305 |                            .size(9.8) | support logic; impact depends on neighboring block and data flow. |
| 5306 |                            .monospace() | support logic; impact depends on neighboring block and data flow. |
| 5307 |                            .color(Color32::from_gray(170)), | support logic; impact depends on neighboring bl
| 5308 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5309 |                    } else { | support logic; impact depends on neighboring block and data flow. |
| 5310 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5311 |                            RichText::new( | support logic; impact depends on neighboring block and data flow.
| 5312 |                                "This selected fault is network/power domain and may not map to an ECM SPN/FMI
| 5313 |                            ) | support logic; impact depends on neighboring block and data flow. |
| 5314 |                            .size(9.8) | support logic; impact depends on neighboring block and data flow. |
| 5315 |                            .color(Color32::from_gray(155)), | support logic; impact depends on neighboring bl
| 5316 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5317 |                    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5318 |                }); | support logic; impact depends on neighboring block and data flow. |
| 5319 |        }); | support logic; impact depends on neighboring block and data flow. |
| 5320 |            }); | support logic; impact depends on neighboring block and data flow. |
| 5321 |    } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5322 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5323 | (blank) | blank separator; no runtime behavior. |
| 5324 | // ═══════════════════════════════════════════════════════════════════════════════ | comment
| 5325 | // TAB BOOT | comment/documentation; changing affects understanding and intent traceability. |
| 5326 | // ═══════════════════════════════════════════════════════════════════════════════ | comment
| 5327 | impl App { | implementation binding; behavior semantics and trait conformance live here. |
| 5328 |     fn tab_boot(&self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flow.
| 5329 |         let ign = self.bench.ignition(); | support logic; impact depends on neighboring block and data flow. |
| 5330 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 5331 |             for (lbl, s) in [ | iteration logic; may alter timing, throughput, and safety constraints. |
| 5332 |                 ("OFF", IgnitionState::Off), | support logic; impact depends on neighboring block and data flow
| 5333 |                 ("ACC", IgnitionState::Accessory), | support logic; impact depends on neighboring block and dat
| 5334 |                 ("ON", IgnitionState::On), | support logic; impact depends on neighboring block and data flow.
| 5335 |                 ("CRANK", IgnitionState::Cranking), | support logic; impact depends on neighboring block and da
| 5336 |                 ("RUN", IgnitionState::Running), | support logic; impact depends on neighboring block and data
| 5337 |             ] { | support logic; impact depends on neighboring block and data flow. |
| 5338 |                 let cur = ign == s; | support logic; impact depends on neighboring block and data flow. |
| 5339 |                 let c = if cur { | support logic; impact depends on neighboring block and data flow. |
| 5340 |                     Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 5341 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 5342 |                     Color32::from_gray(80) | support logic; impact depends on neighboring block and data flow.
| 5343 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5344 |                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5345 |                     RichText::new(format!("[{}]", lbl)) | support logic; impact depends on neighboring block a
| 5346 |                         .size(13.0) | support logic; impact depends on neighboring block and data flow. |
| 5347 |                         .color(c) | support logic; impact depends on neighboring block and data flow. |
| 5348 |                         .strong(), | support logic; impact depends on neighboring block and data flow. |
| 5349 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 5350 |                 if s != IgnitionState::Running { | runtime gate; condition changes can enable/disable critical
| 5351 |                     ui.label(RichText::new("→").size(11.0).color(Color32::from_gray(80))); | support logic; im
| 5352 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5353 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5354 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5355 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5356 |                 RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 5357 |                     "t={:.1}s  Online:{}/{}", | support logic; impact depends on neighboring block and data flo
```

```
| 5358 |                     self.bench.elapsed, | support logic; impact depends on neighboring block and data flow. |
| 5359 |                     self.bench | support logic; impact depends on neighboring block and data flow. |
| 5360 |                         .boot | support logic; impact depends on neighboring block and data flow. |
| 5361 |                         .ecus | support logic; impact depends on neighboring block and data flow. |
| 5362 |                         .iter() | support logic; impact depends on neighboring block and data flow. |
| 5363 |                         .filter(\|e\| e.is_online()) | support logic; impact depends on neighboring block and d
| 5364 |                         .count(), | support logic; impact depends on neighboring block and data flow. |
| 5365 |                     self.bench.boot.ecus.len() | support logic; impact depends on neighboring block and data fl
| 5366 |                 )) | support logic; impact depends on neighboring block and data flow. |
| 5367 |                 .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5368 |                 .color(Color32::from_gray(170)), | support logic; impact depends on neighboring block and data
| 5369 |             ); | support logic; impact depends on neighboring block and data flow. |
| 5370 |         }); | support logic; impact depends on neighboring block and data flow. |
| 5371 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5372 |         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5373 |             RichText::new("SAFETY INTERLOCKS:") | support logic; impact depends on neighboring block and data f
| 5374 |                 .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5375 |                 .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block and data
| 5376 |         ); | support logic; impact depends on neighboring block and data flow. |
| 5377 |         ui.columns(3, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 5378 |             let ils = &self.bench.boot.safety_interlocks; | support logic; impact depends on neighboring block
| 5379 |             let per = ils.len().div_ceil(3); | support logic; impact depends on neighboring block and data flo
| 5380 |             for (ci, col) in cols.iter_mut().enumerate() { | iteration logic; may alter timing, throughput, an
| 5381 |                 for il in ils.iter().skip(ci * per).take(per) { | iteration logic; may alter timing, throughpu
| 5382 |                     let (sym, c) = if il.satisfied { | support logic; impact depends on neighboring block and
| 5383 |                         ("✓", Color32::GREEN) | support logic; impact depends on neighboring block and data fl
| 5384 |                     } else if il.blocks_start { | support logic; impact depends on neighboring block and data f
| 5385 |                         ("✗", Color32::RED) | support logic; impact depends on neighboring block and data flow
| 5386 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 5387 |                         ("■", Color32::YELLOW) | support logic; impact depends on neighboring block and data f
| 5388 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5389 |                     col.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow
| 5390 |                         ui.label(RichText::new(sym).size(13.0).color(c)); | support logic; impact depends on n
| 5391 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5392 |                             RichText::new(format!("{}: {}", il.id, il.description)) | support logic; impact de
| 5393 |                                 .size(10.5) | support logic; impact depends on neighboring block and data flow
| 5394 |                                 .color(Color32::from_gray(185)), | support logic; impact depends on neighborin
| 5395 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5396 |                         if il.blocks_start && !il.satisfied { | runtime gate; condition changes can enable/dis
| 5397 |                             ui.label(RichText::new("[BLOCKS]").size(10.0).color(Color32::RED)); | support logi
| 5398 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5399 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 5400 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5401 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5402 |         }); | support logic; impact depends on neighboring block and data flow. |
| 5403 |         if self.bench.boot.crank_inhibited { | runtime gate; condition changes can enable/disable critical pat
| 5404 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5405 |                 RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 5406 |                     "■ CRANK INHIBITED: {}", | support logic; impact depends on neighboring block and data flo
| 5407 |                     self.bench.boot.crank_inhibit_reason.unwrap_or("?") | error propagation point; changes alt
| 5408 |                 )) | support logic; impact depends on neighboring block and data flow. |
| 5409 |                 .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 5410 |                 .color(Color32::RED), | support logic; impact depends on neighboring block and data flow. |
| 5411 |             ); | support logic; impact depends on neighboring block and data flow. |
| 5412 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 5413 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5414 |                 RichText::new("✓ CRANK CLEAR") | support logic; impact depends on neighboring block and data f
| 5415 |                     .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 5416 |                     .color(Color32::GREEN), | support logic; impact depends on neighboring block and data flow
| 5417 |             ); | support logic; impact depends on neighboring block and data flow. |
| 5418 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5419 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5420 |         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5421 |             RichText::new("BOOT EVENT LOG:") | support logic; impact depends on neighboring block and data flo
| 5422 |                 .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5423 |                 .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block and data
| 5424 |         ); | support logic; impact depends on neighboring block and data flow. |
| 5425 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 5426 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 5427 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 5428 |                 for ev in self.bench.boot.event_log.iter().rev().take(80) { | iteration logic; may alter timing
| 5429 |                     let c = match ev.event { | support logic; impact depends on neighboring block and data flo
| 5430 |                         auto_breaking::boot_sequence::BootEventKind::Running => Color32::GREEN, | support logi
| 5431 |                         auto_breaking::boot_sequence::BootEventKind::Fault => Color32::RED, | support logic; i
| 5432 |                         auto_breaking::boot_sequence::BootEventKind::AddressClaimed => { | support logic; impa
| 5433 |                             Color32::from_rgb(0, 210, 210) | support logic; impact depends on neighboring bloc
```

```
| 5434 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5435 |                        _ => Color32::from_gray(195), | support logic; impact depends on neighboring block and
| 5436 |                    }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5437 |                    ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 5438 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5439 |                            RichText::new(format!("{:7.3}s", ev.timestamp)) | support logic; impact depends on
| 5440 |                                .size(10.5) | support logic; impact depends on neighboring block and data flow
| 5441 |                                .monospace() | support logic; impact depends on neighboring block and data flo
| 5442 |                                .color(Color32::from_gray(115)), | support logic; impact depends on neighborin
| 5443 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5444 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5445 |                            RichText::new(format!("[{:<12}]", ev.ecu_name)) | support logic; impact depends on
| 5446 |                                .size(10.5) | support logic; impact depends on neighboring block and data flow
| 5447 |                                .monospace() | support logic; impact depends on neighboring block and data flo
| 5448 |                                .color(Color32::from_gray(165)), | support logic; impact depends on neighborin
| 5449 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 5450 |                        ui.label(RichText::new(&ev.description).size(10.5).color(c)); | support logic; impact
| 5451 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 5452 |                } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5453 |            }); | support logic; impact depends on neighboring block and data flow. |
| 5454 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5455 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5456 | (blank) | blank separator; no runtime behavior. |
| 5457 | // ═══════════════════════════════════════════════════════════════════════════════ | comment
| 5458 | // TAB IMPLEMENTS — fully interactive | comment/documentation; changing affects understanding and intent trace
| 5459 | // ═══════════════════════════════════════════════════════════════════════════════ | comment
| 5460 | impl App { | implementation binding; behavior semantics and trait conformance live here. |
| 5461 |     fn tab_implements(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and con
| 5462 |         // Snapshot display values | comment/documentation; changing affects understanding and intent traceabi
| 5463 |         let pto_rpm = self.bench.implement.pto_rear_rpm; | support logic; impact depends on neighboring block
| 5464 |         let pto_mode_s = format!("{}", self.bench.implement.pto_mode); | support logic; impact depends on neig
| 5465 |         let pto_target = self.bench.implement.pto_target_rpm; | support logic; impact depends on neighboring b
| 5466 |         let pto_slip = self.bench.implement.pto_slip_pct; | support logic; impact depends on neighboring block
| 5467 |         let pto_torq = self.bench.implement.pto_shaft_torque_nm; | support logic; impact depends on neighborin
| 5468 |         let pto_over = self.bench.implement.overload_protection_active; | support logic; impact depends on nei
| 5469 |         let _pto_en = self.bench.implement.pto_rear_enabled; | support logic; impact depends on neighboring bl
| 5470 |         let hitch_pos = self.bench.implement.hitch_position_pct; | support logic; impact depends on neighborin
| 5471 |         let hitch_draft = self.bench.implement.hitch_draft_force_kn; | support logic; impact depends on neighb
| 5472 |         let hitch_mode = format!("{}", self.bench.implement.hitch_control_mode); | support logic; impact depen
| 5473 |         let hitch_gp = self.bench.implement.hitch_ground_pressure_kpa; | support logic; impact depends on neig
| 5474 |         let hcm_sys_p = self.bench.hcm.system_pressure_bar; | support logic; impact depends on neighboring blo
| 5475 |         let hcm_flow = self.bench.hcm.pump_flow_lpm; | support logic; impact depends on neighboring block and
| 5476 |         let hcm_temp = self.bench.hcm.fluid_temp_c; | support logic; impact depends on neighboring block and d
| 5477 |         let hcm_mode = format!("{}", self.bench.hcm.pump_mode); | support logic; impact depends on neighboring
| 5478 |         let hcm_pwr = self.bench.hcm.hydraulic_power_kw; | support logic; impact depends on neighboring block
| 5479 |         let aux_flows: Vec<f64> = self | support logic; impact depends on neighboring block and data flow. |
| 5480 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 5481 |             .implement | support logic; impact depends on neighboring block and data flow. |
| 5482 |             .aux_banks | support logic; impact depends on neighboring block and data flow. |
| 5483 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 5484 |             .map(\|b\| b.flow_lpm) | support logic; impact depends on neighboring block and data flow. |
| 5485 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 5486 |         let aux_pres: Vec<f64> = self | support logic; impact depends on neighboring block and data flow. |
| 5487 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 5488 |             .implement | support logic; impact depends on neighboring block and data flow. |
| 5489 |             .aux_banks | support logic; impact depends on neighboring block and data flow. |
| 5490 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 5491 |             .map(\|b\| b.pressure_bar) | support logic; impact depends on neighboring block and data flow. |
| 5492 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 5493 |         let aux_dirs: Vec<String> = self | support logic; impact depends on neighboring block and data flow. |
| 5494 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 5495 |             .implement | support logic; impact depends on neighboring block and data flow. |
| 5496 |             .aux_banks | support logic; impact depends on neighboring block and data flow. |
| 5497 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 5498 |             .map(\|b\| format!("{}", b.direction)) | support logic; impact depends on neighboring block and da
| 5499 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 5500 |         let loader_lift = self.bench.hcm.loader_lift.position_pct(); | support logic; impact depends on neighb
| 5501 |         let _loader_tilt = self.bench.hcm.loader_tilt.position_pct(); | support logic; impact depends on neigh
| 5502 | (blank) | blank separator; no runtime behavior. |
| 5503 |         ui.columns(3, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 5504 |             // ══ PTO ═══════════════════════════════════════════════════════ | comment/doc
| 5505 |             let ui = &mut cols[0]; | support logic; impact depends on neighboring block and data flow. |
| 5506 |             egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data flow.
| 5507 |                 .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data fl
| 5508 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 5509 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
```

```
| 5510 |                         RichText::new("POWER TAKE-OFF (TDP)") | support logic; impact depends on neighboring bl
| 5511 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 5512 |                             .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighbo
| 5513 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5514 |                     egui::Frame::dark_canvas(ui.style()).show(ui, \|ui\| { | support logic; impact depends on n
| 5515 |                         arc_gauge( | support logic; impact depends on neighboring block and data flow. |
| 5516 |                             ui, pto_rpm, 0.0, 1100.0, "PTO REAR", "rpm", 140.0, 900.0, 1050.0, None, | support
| 5517 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5518 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 5519 |                     digital_readout(ui, "Mode", &pto_mode_s, Color32::YELLOW); | support logic; impact depends
| 5520 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 5521 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 5522 |                         "Target", | support logic; impact depends on neighboring block and data flow. |
| 5523 |                         &format!("{:.0} rpm", pto_target), | support logic; impact depends on neighboring bloc
| 5524 |                         Color32::from_gray(185), | support logic; impact depends on neighboring block and data
| 5525 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5526 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 5527 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 5528 |                         "Shaft Tq", | support logic; impact depends on neighboring block and data flow. |
| 5529 |                         &format!("{:.0} Nm", pto_torq), | support logic; impact depends on neighboring block a
| 5530 |                         Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring block a
| 5531 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5532 |                     bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 5533 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 5534 |                         "Slip", | support logic; impact depends on neighboring block and data flow. |
| 5535 |                         pto_slip, | support logic; impact depends on neighboring block and data flow. |
| 5536 |                         100.0, | support logic; impact depends on neighboring block and data flow. |
| 5537 |                         "%", | support logic; impact depends on neighboring block and data flow. |
| 5538 |                         120.0, | support logic; impact depends on neighboring block and data flow. |
| 5539 |                         if pto_slip > 10.0 { | runtime gate; condition changes can enable/disable critical pat
| 5540 |                             Color32::YELLOW | support logic; impact depends on neighboring block and data flow
| 5541 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5542 |                             Color32::GREEN | support logic; impact depends on neighboring block and data flow.
| 5543 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 5544 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5545 |                     if pto_over { | runtime gate; condition changes can enable/disable critical paths. |
| 5546 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5547 |                             RichText::new("■ OVERLOAD!") | support logic; impact depends on neighboring block
| 5548 |                                 .size(12.0) | support logic; impact depends on neighboring block and data flow
| 5549 |                                 .color(Color32::RED) | support logic; impact depends on neighboring block and
| 5550 |                                 .strong(), | support logic; impact depends on neighboring block and data flow.
| 5551 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5552 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5553 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5554 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5555 |                         RichText::new("Mode select:") | support logic; impact depends on neighboring block and
| 5556 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5557 |                             .color(Color32::from_gray(140)), | support logic; impact depends on neighboring bl
| 5558 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5559 |                     ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and da
| 5560 |                         for (lbl, mode) in [ | iteration logic; may alter timing, throughput, and safety const
| 5561 |                             ("OFF", PtoMode::Off), | support logic; impact depends on neighboring block and da
| 5562 |                             ("540", PtoMode::Std540), | support logic; impact depends on neighboring block and
| 5563 |                             ("1000", PtoMode::Std1000), | support logic; impact depends on neighboring block a
| 5564 |                             ("540eco", PtoMode::Economy540), | support logic; impact depends on neighboring bl
| 5565 |                             ("1000eco", PtoMode::Economy1000), | support logic; impact depends on neighboring
| 5566 |                         ] { | support logic; impact depends on neighboring block and data flow. |
| 5567 |                             let active = pto_mode_s.trim() == format!("{}", mode).trim(); | support logic; impa
| 5568 |                             let fill = if active { | support logic; impact depends on neighboring block and da
| 5569 |                                 Color32::from_rgb(20, 80, 20) | support logic; impact depends on neighboring b
| 5570 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 5571 |                                 Color32::from_gray(35) | support logic; impact depends on neighboring block an
| 5572 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 5573 |                             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 5574 |                                 .add(Button::new(RichText::new(lbl).size(11.0)).fill(fill)) | support logic; i
| 5575 |                                 .clicked() | support logic; impact depends on neighboring block and data flow.
| 5576 |                             { | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 5577 |                                 self.cmds.push(Cmd::SetPtoMode(mode)); | support logic; impact depends on neig
| 5578 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 5579 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5580 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 5581 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 5582 | (blank) | blank separator; no runtime behavior. |
| 5583 |             // ■■ 3-Point Hitch ■■■
```

```
| 5586 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 5587 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5588 |                         RichText::new("3-POINT HITCH") | support logic; impact depends on neighboring block and
| 5589 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 5590 |                             .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighbo
| 5591 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5592 |                     // Visual position bar | comment/documentation; changing affects understanding and intent
| 5593 |                     let (rect, _) = ui | support logic; impact depends on neighboring block and data flow. |
| 5594 |                         .allocate_exact_size(Vec2::new(ui.available_width(), 70.0), Sense::hover()); | support
| 5595 |                     let p = ui.painter_at(rect); | support logic; impact depends on neighboring block and data
| 5596 |                     p.rect_filled(rect, 4.0, Color32::from_gray(20)); | support logic; impact depends on neigh
| 5597 |                     let frac = 1.0 - hitch_pos as f32 / 100.0; | support logic; impact depends on neighboring
| 5598 |                     let fill_h = rect.height() * frac; | support logic; impact depends on neighboring block and
| 5599 |                     let fill_r = Rect::from_min_size( | support logic; impact depends on neighboring block and
| 5600 |                         Pos2::new(rect.min.x, rect.min.y + fill_h), | support logic; impact depends on neighbo
| 5601 |                         Vec2::new(rect.width(), rect.height() - fill_h), | support logic; impact depends on ne
| 5602 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5603 |                     p.rect_filled(fill_r, 4.0, Color32::from_rgb(30, 140, 200)); | support logic; impact depen
| 5604 |                     p.text( | support logic; impact depends on neighboring block and data flow. |
| 5605 |                         rect.center(), | support logic; impact depends on neighboring block and data flow. |
| 5606 |                         Align2::CENTER_CENTER, | support logic; impact depends on neighboring block and data f
| 5607 |                         format!("{:.1}%", hitch_pos), | support logic; impact depends on neighboring block and
| 5608 |                         FontId::proportional(18.0), | support logic; impact depends on neighboring block and d
| 5609 |                         Color32::WHITE, | support logic; impact depends on neighboring block and data flow. |
| 5610 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5611 | (blank) | blank separator; no runtime behavior. |
| 5612 |                     bar_gauge(ui, "Draft", hitch_draft, 50.0, "kN", 120.0, Color32::YELLOW); | support logic;
| 5613 |                     digital_readout(ui, "Mode", &hitch_mode, Color32::from_gray(200)); | support logic; impact
| 5614 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 5615 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 5616 |                         "Gnd P", | support logic; impact depends on neighboring block and data flow. |
| 5617 |                         &format!("{:.1} kPa", hitch_gp), | support logic; impact depends on neighboring block
| 5618 |                         Color32::from_gray(180), | support logic; impact depends on neighboring block and data
| 5619 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5620 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5621 |                     // Target slider | comment/documentation; changing affects understanding and intent traceab
| 5622 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5623 |                         RichText::new("Target position:") | support logic; impact depends on neighboring block
| 5624 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5625 |                             .color(Color32::from_gray(140)), | support logic; impact depends on neighboring bl
| 5626 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5627 |                     let mut ht = self.hitch_target as f32; | support logic; impact depends on neighboring bloc
| 5628 |                     ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 5629 |                         [ui.available_width() - 10.0, 22.0], | support logic; impact depends on neighboring bl
| 5630 |                         Slider::new(&mut ht, 0.0..=100.0).suffix(" %"), | support logic; impact depends on nei
| 5631 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5632 |                     if ht as f64 != self.hitch_target { | runtime gate; condition changes can enable/disable c
| 5633 |                         self.cmds.push(Cmd::SetHitchTarget(ht as f64)); | support logic; impact depends on nei
| 5634 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5635 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 5636 |                         if ui.button("▲ RAISE").clicked() { | runtime gate; condition changes can enable/disab
| 5637 |                             self.cmds.push(Cmd::SetHitchJoystick(1.0)); | support logic; impact depends on nei
| 5638 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5639 |                         if ui.button("■ HOLD").clicked() { | runtime gate; condition changes can enable/disabl
| 5640 |                             self.cmds.push(Cmd::SetHitchJoystick(0.0)); | support logic; impact depends on nei
| 5641 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5642 |                         if ui.button("▼ LOWER").clicked() { | runtime gate; condition changes can enable/disab
| 5643 |                             self.cmds.push(Cmd::SetHitchJoystick(-1.0)); | support logic; impact depends on ne
| 5644 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5645 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 5646 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 5647 | (blank) | blank separator; no runtime behavior. |
| 5648 |             // ■■ Hydraulics + Aux Valves ■■■
```

```
| 5662 |                         "Flow", | support logic; impact depends on neighboring block and data flow. |
| 5663 |                         hcm_flow, | support logic; impact depends on neighboring block and data flow. |
| 5664 |                         300.0, | support logic; impact depends on neighboring block and data flow. |
| 5665 |                         "L/m", | support logic; impact depends on neighboring block and data flow. |
| 5666 |                         w, | support logic; impact depends on neighboring block and data flow. |
| 5667 |                         Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring block an
| 5668 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5669 |                     bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 5670 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 5671 |                         "Temp", | support logic; impact depends on neighboring block and data flow. |
| 5672 |                         hcm_temp, | support logic; impact depends on neighboring block and data flow. |
| 5673 |                         120.0, | support logic; impact depends on neighboring block and data flow. |
| 5674 |                         "°C", | support logic; impact depends on neighboring block and data flow. |
| 5675 |                         w, | support logic; impact depends on neighboring block and data flow. |
| 5676 |                         if hcm_temp > 90.0 { | runtime gate; condition changes can enable/disable critical pat
| 5677 |                             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 5678 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5679 |                             Color32::from_rgb(80, 180, 255) | support logic; impact depends on neighboring bloc
| 5680 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 5681 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5682 |                     digital_readout(ui, "Mode", &hcm_mode, Color32::from_gray(200)); | support logic; impact de
| 5683 |                     digital_readout(ui, "Power", &format!("{:.1} kW", hcm_pwr), Color32::YELLOW); | support log
| 5684 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5685 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5686 |                         RichText::new("LOADER:") | support logic; impact depends on neighboring block and data
| 5687 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5688 |                             .color(Color32::from_gray(140)), | support logic; impact depends on neighboring bl
| 5689 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5690 |                     // Loader lift slider | comment/documentation; changing affects understanding and intent t
| 5691 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 5692 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5693 |                             RichText::new("Lift") | support logic; impact depends on neighboring block and data
| 5694 |                                 .size(11.0) | support logic; impact depends on neighboring block and data flow
| 5695 |                                 .color(Color32::from_gray(150)), | support logic; impact depends on neighborin
| 5696 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5697 |                         let fill_w = (loader_lift as f32 / 100.0 * 80.0).max(2.0); | support logic; impact dep
| 5698 |                         let (bar, _) = | support logic; impact depends on neighboring block and data flow. |
| 5699 |                             ui.allocate_exact_size(Vec2::new(80.0, 14.0), Sense::hover()); | support logic; im
| 5700 |                         let p = ui.painter_at(bar); | support logic; impact depends on neighboring block and d
| 5701 |                         p.rect_filled(bar, 3.0, Color32::from_gray(22)); | support logic; impact depends on ne
| 5702 |                         p.rect_filled( | support logic; impact depends on neighboring block and data flow. |
| 5703 |                             Rect::from_min_size(bar.min, Vec2::new(fill_w, bar.height())), | support logic; im
| 5704 |                             3.0, | support logic; impact depends on neighboring block and data flow. |
| 5705 |                             Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighboring bl
| 5706 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5707 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5708 |                             RichText::new(format!("{:.0}%", loader_lift)) | support logic; impact depends on n
| 5709 |                                 .size(10.5) | support logic; impact depends on neighboring block and data flow
| 5710 |                                 .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighboring bl
| 5711 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5712 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 5713 |                     let mut ll = self.bench.loader_lift_cmd as f32; | support logic; impact depends on neighbo
| 5714 |                     ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 5715 |                         [ui.available_width() - 10.0, 20.0], | support logic; impact depends on neighboring bl
| 5716 |                         Slider::new(&mut ll, -1.0..=1.0).text("Lift cmd"), | support logic; impact depends on
| 5717 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5718 |                     if ll as f64 != self.bench.loader_lift_cmd { | runtime gate; condition changes can enable/
| 5719 |                         self.cmds.push(Cmd::SetLoaderLift(ll as f64)); | support logic; impact depends on neig
| 5720 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5721 |                     let mut lt = self.bench.loader_tilt_cmd as f32; | support logic; impact depends on neighbo
| 5722 |                     ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 5723 |                         [ui.available_width() - 10.0, 20.0], | support logic; impact depends on neighboring bl
| 5724 |                         Slider::new(&mut lt, -1.0..=1.0).text("Tilt cmd"), | support logic; impact depends on
| 5725 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5726 |                     if lt as f64 != self.bench.loader_tilt_cmd { | runtime gate; condition changes can enable/
| 5727 |                         self.cmds.push(Cmd::SetLoaderTilt(lt as f64)); | support logic; impact depends on neig
| 5728 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5729 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5730 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5731 |                         RichText::new("AUX HYDRAULIC VALVES:") | support logic; impact depends on neighboring
| 5732 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5733 |                             .color(Color32::from_gray(140)), | support logic; impact depends on neighboring bl
| 5734 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5735 |                     for i in 0..4 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 5736 |                         let cmd_val = self.aux_cmds[i]; | support logic; impact depends on neighboring block a
| 5737 |                         ui.push_id(format!("impl::aux::{}", i), \|ui\| { | support logic; impact depends on ne
```

```
| 5738 |                             ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and dat
| 5739 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 5740 |                                     RichText::new(format!("Bank{}:", i)) | support logic; impact depends on nei
| 5741 |                                         .size(11.0) | support logic; impact depends on neighboring block and da
| 5742 |                                         .color(Color32::from_gray(155)), | support logic; impact depends on nei
| 5743 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 5744 |                                 // - button | comment/documentation; changing affects understanding and intent
| 5745 |                                 if ui.small_button("■").clicked() { | runtime gate; condition changes can enab
| 5746 |                                     self.cmds.push(Cmd::SetAuxValve(i, -1.0)); | support logic; impact depends
| 5747 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 5748 |                                 if ui.small_button("■").clicked() { | runtime gate; condition changes can enab
| 5749 |                                     self.cmds.push(Cmd::SetAuxValve(i, 0.0)); | support logic; impact depends o
| 5750 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 5751 |                                 if ui.small_button("■").clicked() { | runtime gate; condition changes can enab
| 5752 |                                     self.cmds.push(Cmd::SetAuxValve(i, 1.0)); | support logic; impact depends o
| 5753 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 5754 |                                 let c = if cmd_val.abs() > 0.05 { | support logic; impact depends on neighborin
| 5755 |                                     Color32::GREEN | support logic; impact depends on neighboring block and dat
| 5756 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 5757 |                                     Color32::from_gray(80) | support logic; impact depends on neighboring bloc
| 5758 |                                 }; | scope delimiter; structural only, but wrong placement changes ownership/li
| 5759 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 5760 |                                     RichText::new(format!( | support logic; impact depends on neighboring bloc
| 5761 |                                         "{} {:.0}L/m {:.0}bar", | support logic; impact depends on neighboring
| 5762 |                                         aux_dirs[i], aux_flows[i], aux_pres[i] | support logic; impact depends
| 5763 |                                     )) | support logic; impact depends on neighboring block and data flow. |
| 5764 |                                     .size(10.5) | support logic; impact depends on neighboring block and data f
| 5765 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 5766 |                                     .color(c), | support logic; impact depends on neighboring block and data fl
| 5767 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 5768 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 5769 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 5770 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5771 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 5772 | (blank) | blank separator; no runtime behavior. |
| 5773 |             // ■■ Implement info ■■■
```

```
| 5814 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5815 |                             Color32::from_gray(100) | support logic; impact depends on neighboring block and da
| 5816 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 5817 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5818 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5819 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5820 |                         RichText::new("FRONT PTO:") | support logic; impact depends on neighboring block and da
| 5821 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5822 |                             .color(Color32::from_gray(140)), | support logic; impact depends on neighboring blo
| 5823 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5824 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 5825 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 5826 |                         "Front RPM", | support logic; impact depends on neighboring block and data flow. |
| 5827 |                         &format!("{:.0}", imp.pto_front_rpm), | support logic; impact depends on neighboring bl
| 5828 |                         if imp.pto_front_enabled { | runtime gate; condition changes can enable/disable critica
| 5829 |                             Color32::GREEN | support logic; impact depends on neighboring block and data flow.
| 5830 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5831 |                             Color32::from_gray(100) | support logic; impact depends on neighboring block and da
| 5832 |                         }, | support logic; impact depends on neighboring block and data flow. |
| 5833 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5834 |                     let fen = imp.pto_front_enabled; | support logic; impact depends on neighboring block and c
| 5835 |                     if ui | runtime gate; condition changes can enable/disable critical paths. |
| 5836 |                         .add( | support logic; impact depends on neighboring block and data flow. |
| 5837 |                             Button::new( | support logic; impact depends on neighboring block and data flow. |
| 5838 |                                 RichText::new(if fen { "Front PTO ON" } else { "Front PTO OFF" }) | support log
| 5839 |                                     .size(11.0), | support logic; impact depends on neighboring block and data
| 5840 |                             ) | support logic; impact depends on neighboring block and data flow. |
| 5841 |                             .fill(if fen { | support logic; impact depends on neighboring block and data flow.
| 5842 |                                 Color32::from_rgb(10, 80, 10) | support logic; impact depends on neighboring bl
| 5843 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 5844 |                                 Color32::from_gray(35) | support logic; impact depends on neighboring block and
| 5845 |                             }), | support logic; impact depends on neighboring block and data flow. |
| 5846 |                         ) | support logic; impact depends on neighboring block and data flow. |
| 5847 |                         .clicked() | support logic; impact depends on neighboring block and data flow. |
| 5848 |                     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5849 |                         self.bench.implement.pto_front_enabled = !fen; | support logic; impact depends on neigh
| 5850 |                         self.implement_feedback = if fen { | support logic; impact depends on neighboring bloc
| 5851 |                             "Front PTO -> OFF".into() | support logic; impact depends on neighboring block and
| 5852 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5853 |                             "Front PTO -> ON".into() | support logic; impact depends on neighboring block and c
| 5854 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 5855 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5856 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5857 |                     // Hitch control mode | comment/documentation; changing affects understanding and intent t
| 5858 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5859 |                         RichText::new("HITCH CONTROL MODE:") | support logic; impact depends on neighboring blo
| 5860 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 5861 |                             .color(Color32::from_gray(140)), | support logic; impact depends on neighboring blo
| 5862 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5863 |                     for (mode_s, mode) in [ | iteration logic; may alter timing, throughput, and safety constra
| 5864 |                         ("Position", &auto_breaking::implement::HitchMode::Position), | support logic; impact d
| 5865 |                         ("Draft", &auto_breaking::implement::HitchMode::Draft), | support logic; impact depends
| 5866 |                         ("Mixed", &auto_breaking::implement::HitchMode::Mixed), | support logic; impact depends
| 5867 |                         ("Float", &auto_breaking::implement::HitchMode::Float), | support logic; impact depends
| 5868 |                     ] { | support logic; impact depends on neighboring block and data flow. |
| 5869 |                         let cur = hitch_mode.trim() == format!("{}", mode).trim(); | support logic; impact depe
| 5870 |                         let fill = if cur { | support logic; impact depends on neighboring block and data flow
| 5871 |                             Color32::from_rgb(25, 75, 25) | support logic; impact depends on neighboring block
| 5872 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 5873 |                             Color32::from_gray(35) | support logic; impact depends on neighboring block and dat
| 5874 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 5875 |                         if ui | runtime gate; condition changes can enable/disable critical paths. |
| 5876 |                             .add( | support logic; impact depends on neighboring block and data flow. |
| 5877 |                                 Button::new(RichText::new(mode_s).size(11.0)) | support logic; impact depends o
| 5878 |                                     .fill(fill) | support logic; impact depends on neighboring block and data f
| 5879 |                                     .min_size(Vec2::new(90.0, 22.0)), | support logic; impact depends on neigh
| 5880 |                             ) | support logic; impact depends on neighboring block and data flow. |
| 5881 |                             .clicked() | support logic; impact depends on neighboring block and data flow. |
| 5882 |                         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5883 |                             self.bench.implement.hitch_control_mode = *mode; | support logic; impact depends on
| 5884 |                             self.implement_feedback = | support logic; impact depends on neighboring block and
| 5885 |                                 format!("Hitch control mode -> {}", mode_s); | support logic; impact depends on
| 5886 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 5887 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5888 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 5889 |                     let hitch_err = (self.hitch_target - hitch_pos).abs(); | support logic; impact depends on n
```

```
| 5890 |                     let hitch_col = if hitch_err > 12.0 { | support logic; impact depends on neighboring block
| 5891 |                         Color32::YELLOW | support logic; impact depends on neighboring block and data flow. |
| 5892 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 5893 |                         Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 5894 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5895 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5896 |                         RichText::new(format!( | support logic; impact depends on neighboring block and data f
| 5897 |                             "Control tracking: Hitch target {:.1}% / actual {:.1}% (err {:.1}%)", | support lo
| 5898 |                             self.hitch_target, hitch_pos, hitch_err | support logic; impact depends on neighbo
| 5899 |                         )) | support logic; impact depends on neighboring block and data flow. |
| 5900 |                         .size(10.2) | support logic; impact depends on neighboring block and data flow. |
| 5901 |                         .color(hitch_col), | support logic; impact depends on neighboring block and data flow.
| 5902 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 5903 |                     if !self.implement_feedback.is_empty() { | runtime gate; condition changes can enable/disa
| 5904 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 5905 |                             RichText::new(format!("Last command: {}", self.implement_feedback)) | support logi
| 5906 |                                 .size(10.2) | support logic; impact depends on neighboring block and data flow
| 5907 |                                 .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighboring bl
| 5908 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 5909 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5910 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 5911 |         }); | support logic; impact depends on neighboring block and data flow. |
| 5912 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5913 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 5914 | (blank) | blank separator; no runtime behavior. |
| 5915 | // ━━━
```

```
| 5966 |                             if changed { | runtime gate; condition changes can enable/disable critical paths.
| 5967 |                                 self.cmds.push(Cmd::SetFuelLevel(self.p_fuel)); | support logic; impact depends
| 5968 |                                 self.params_dirty = true; | support logic; impact depends on neighboring block
| 5969 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 5970 | (blank) | blank separator; no runtime behavior. |
| 5971 |                             let (mut v, mut changed) = (self.p_def as f32, false); | support logic; impact depe
| 5972 |                             ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and dat
| 5973 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 5974 |                                     RichText::new("DEF / AdBlue %") | support logic; impact depends on neighbor
| 5975 |                                         .size(11.0) | support logic; impact depends on neighboring block and da
| 5976 |                                         .color(Color32::from_rgb(0, 210, 210)), | support logic; impact depends
| 5977 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 5978 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 5979 |                             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 5980 |                                 .add_sized( | support logic; impact depends on neighboring block and data flow
| 5981 |                                     [ui.available_width() - 10.0, 22.0], | support logic; impact depends on nei
| 5982 |                                     Slider::new(&mut v, 0.0..=100.0).suffix(" %"), | support logic; impact depe
| 5983 |                                 ) | support logic; impact depends on neighboring block and data flow. |
| 5984 |                                 .changed() | support logic; impact depends on neighboring block and data flow.
| 5985 |                             { | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 5986 |                                 self.p_def = v as f64; | support logic; impact depends on neighboring block and
| 5987 |                                 changed = true; | support logic; impact depends on neighboring block and data
| 5988 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 5989 |                             if changed { | runtime gate; condition changes can enable/disable critical paths.
| 5990 |                                 self.cmds.push(Cmd::SetDefLevel(self.p_def)); | support logic; impact depends
| 5991 |                                 self.params_dirty = true; | support logic; impact depends on neighboring block
| 5992 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 5993 | (blank) | blank separator; no runtime behavior. |
| 5994 |                             let (mut v, mut changed) = (self.p_engine_h as f32, false); | support logic; impact
| 5995 |                             ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and dat
| 5996 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 5997 |                                     RichText::new("Engine Hours") | support logic; impact depends on neighborin
| 5998 |                                         .size(11.0) | support logic; impact depends on neighboring block and da
| 5999 |                                         .color(Color32::from_gray(200)), | support logic; impact depends on nei
| 6000 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 6001 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 6002 |                             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6003 |                                 .add_sized( | support logic; impact depends on neighboring block and data flow
| 6004 |                                     [ui.available_width() - 10.0, 22.0], | support logic; impact depends on nei
| 6005 |                                     Slider::new(&mut v, 0.0..=100000.0).suffix(" h"), | support logic; impact
| 6006 |                                 ) | support logic; impact depends on neighboring block and data flow. |
| 6007 |                                 .changed() | support logic; impact depends on neighboring block and data flow.
| 6008 |                             { | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6009 |                                 self.p_engine_h = v as f64; | support logic; impact depends on neighboring bloc
| 6010 |                                 changed = true; | support logic; impact depends on neighboring block and data
| 6011 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6012 |                             if changed { | runtime gate; condition changes can enable/disable critical paths.
| 6013 |                                 self.cmds.push(Cmd::SetEngineHours(self.p_engine_h)); | support logic; impact
| 6014 |                                 self.params_dirty = true; | support logic; impact depends on neighboring block
| 6015 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6016 | (blank) | blank separator; no runtime behavior. |
| 6017 |                             ui.add_space(8.0); | support logic; impact depends on neighboring block and data f
| 6018 |                             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6019 |                                 .add( | support logic; impact depends on neighboring block and data flow. |
| 6020 |                                     Button::new(RichText::new("■ Sync from simulation").size(11.0)) | support
| 6021 |                                         .fill(Color32::from_gray(40)), | support logic; impact depends on neigh
| 6022 |                                 ) | support logic; impact depends on neighboring block and data flow. |
| 6023 |                                 .clicked() | support logic; impact depends on neighboring block and data flow.
| 6024 |                             { | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6025 |                                 self.params_dirty = false; | support logic; impact depends on neighboring bloc
| 6026 |                             } // triggers sync_params_from_bench next tick | support logic; impact depends on
| 6027 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 6028 | (blank) | blank separator; no runtime behavior. |
| 6029 |                     // ■■ Temperatures ■■■
```

```
| 6042 |                                RichText::new("Coolant Temp °C") | support logic; impact depends on neighboring
| 6043 |                                    .size(11.0) | support logic; impact depends on neighboring block and data f
| 6044 |                                    .color(Color32::from_rgb(255, 130, 30)), | support logic; impact depends on
| 6045 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 6046 |                            if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6047 |                                .add_sized( | support logic; impact depends on neighboring block and data flow
| 6048 |                                    [ui.available_width() - 10.0, 22.0], | support logic; impact depends on nei
| 6049 |                                    Slider::new(&mut v, -40.0..=130.0).suffix(" °C"), | support logic; impact
| 6050 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 6051 |                                .changed() | support logic; impact depends on neighboring block and data flow.
| 6052 |                            { | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6053 |                                self.p_coolant = v as f64; | support logic; impact depends on neighboring block
| 6054 |                                changed = true; | support logic; impact depends on neighboring block and data f
| 6055 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6056 |                            if changed { | runtime gate; condition changes can enable/disable critical paths.
| 6057 |                                self.cmds.push(Cmd::SetCoolantTemp(self.p_coolant)); | support logic; impact de
| 6058 |                                self.params_dirty = true; | support logic; impact depends on neighboring block
| 6059 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6060 | (blank) | blank separator; no runtime behavior. |
| 6061 |                            let (mut v, mut changed) = (self.p_oil_prs as f32, false); | support logic; impact
| 6062 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6063 |                                RichText::new("Oil Pressure kPa") | support logic; impact depends on neighborin
| 6064 |                                    .size(11.0) | support logic; impact depends on neighboring block and data f
| 6065 |                                    .color(Color32::GREEN), | support logic; impact depends on neighboring bloc
| 6066 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 6067 |                            if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6068 |                                .add_sized( | support logic; impact depends on neighboring block and data flow
| 6069 |                                    [ui.available_width() - 10.0, 22.0], | support logic; impact depends on nei
| 6070 |                                    Slider::new(&mut v, 0.0..=800.0).suffix(" kPa"), | support logic; impact de
| 6071 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 6072 |                                .changed() | support logic; impact depends on neighboring block and data flow.
| 6073 |                            { | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6074 |                                self.p_oil_prs = v as f64; | support logic; impact depends on neighboring block
| 6075 |                                changed = true; | support logic; impact depends on neighboring block and data f
| 6076 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6077 |                            if changed { | runtime gate; condition changes can enable/disable critical paths.
| 6078 |                                self.cmds.push(Cmd::SetOilPressure(self.p_oil_prs)); | support logic; impact de
| 6079 |                                self.params_dirty = true; | support logic; impact depends on neighboring block
| 6080 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6081 | (blank) | blank separator; no runtime behavior. |
| 6082 |                            let (mut v, mut changed) = (self.p_boost as f32, false); | support logic; impact de
| 6083 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6084 |                                RichText::new("Boost Pressure kPa") | support logic; impact depends on neighbor
| 6085 |                                    .size(11.0) | support logic; impact depends on neighboring block and data f
| 6086 |                                    .color(Color32::from_rgb(180, 80, 255)), | support logic; impact depends on
| 6087 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 6088 |                            if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6089 |                                .add_sized( | support logic; impact depends on neighboring block and data flow
| 6090 |                                    [ui.available_width() - 10.0, 22.0], | support logic; impact depends on nei
| 6091 |                                    Slider::new(&mut v, 0.0..=350.0).suffix(" kPa"), | support logic; impact de
| 6092 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 6093 |                                .changed() | support logic; impact depends on neighboring block and data flow.
| 6094 |                            { | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6095 |                                self.p_boost = v as f64; | support logic; impact depends on neighboring block a
| 6096 |                                changed = true; | support logic; impact depends on neighboring block and data f
| 6097 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6098 |                            if changed { | runtime gate; condition changes can enable/disable critical paths.
| 6099 |                                self.cmds.push(Cmd::SetBoostPressure(self.p_boost)); | support logic; impact de
| 6100 |                                self.params_dirty = true; | support logic; impact depends on neighboring block
| 6101 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6102 | (blank) | blank separator; no runtime behavior. |
| 6103 |                            ui.add_space(8.0); | support logic; impact depends on neighboring block and data fl
| 6104 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6105 |                                RichText::new( | support logic; impact depends on neighboring block and data fl
| 6106 |                                    "Simulate sensor faults by moving sliders to extreme values.", | support lo
| 6107 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 6108 |                                .size(10.0) | support logic; impact depends on neighboring block and data flow
| 6109 |                                .color(Color32::from_gray(140)) | support logic; impact depends on neighboring
| 6110 |                                .italics(), | support logic; impact depends on neighboring block and data flow
| 6111 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 6112 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 6113 | (blank) | blank separator; no runtime behavior. |
| 6114 |                    // ■■ Transmission & BCM ■■■
```

```
| 6118 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6119 |                                    RichText::new("TRANSMISSION CONTROLS") | support logic; impact depends on neigh
| 6120 |                                        .size(12.0) | support logic; impact depends on neighboring block and data f
| 6121 |                                        .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 6122 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 6123 |                                ui.separator(); | support logic; impact depends on neighboring block and data flow
| 6124 |                                ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and da
| 6125 |                                    if ui.button("■ DN").clicked() { | runtime gate; condition changes can enable/
| 6126 |                                        self.cmds.push(Cmd::ManualDn); | support logic; impact depends on neighbor
| 6127 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6128 |                                    if ui.button("■ UP").clicked() { | runtime gate; condition changes can enable/
| 6129 |                                        self.cmds.push(Cmd::ManualUp); | support logic; impact depends on neighbor
| 6130 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6131 |                                    if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6132 |                                        .add(Button::new(RichText::new("AUTO").size(11.0)).fill( | support logic;
| 6133 |                                            if self.bench.tcm.auto_mode == AutoShiftMode::Auto { | runtime gate; c
| 6134 |                                                Color32::from_rgb(20, 80, 20) | support logic; impact depends on n
| 6135 |                                            } else { | support logic; impact depends on neighboring block and data
| 6136 |                                                Color32::from_gray(35) | support logic; impact depends on neighbor
| 6137 |                                            }, | support logic; impact depends on neighboring block and data flow.
| 6138 |                                        )) | support logic; impact depends on neighboring block and data flow. |
| 6139 |                                        .clicked() | support logic; impact depends on neighboring block and data f
| 6140 |                                    { | scope delimiter; structural only, but wrong placement changes ownership/li
| 6141 |                                        self.cmds.push(Cmd::ToggleAutoShift); | support logic; impact depends on n
| 6142 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6143 |                                    if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6144 |                                        .add(Button::new(RichText::new("CREEP").size(11.0)).fill( | support logic;
| 6145 |                                            if self.bench.tcm.creeper_engaged { | runtime gate; condition changes
| 6146 |                                                Color32::from_rgb(80, 50, 10) | support logic; impact depends on n
| 6147 |                                            } else { | support logic; impact depends on neighboring block and data
| 6148 |                                                Color32::from_gray(35) | support logic; impact depends on neighbor
| 6149 |                                            }, | support logic; impact depends on neighboring block and data flow.
| 6150 |                                        )) | support logic; impact depends on neighboring block and data flow. |
| 6151 |                                        .clicked() | support logic; impact depends on neighboring block and data f
| 6152 |                                    { | scope delimiter; structural only, but wrong placement changes ownership/li
| 6153 |                                        self.cmds.push(Cmd::ToggleCreeper); | support logic; impact depends on neig
| 6154 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6155 |                                }); | support logic; impact depends on neighboring block and data flow. |
| 6156 |                                ui.separator(); | support logic; impact depends on neighboring block and data flow
| 6157 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6158 |                                    RichText::new("BCM CONTROLS") | support logic; impact depends on neighboring b
| 6159 |                                        .size(12.0) | support logic; impact depends on neighboring block and data f
| 6160 |                                        .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 6161 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 6162 |                                ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring bloc
| 6163 |                                    if ui.button("Work Lights").clicked() { | runtime gate; condition changes can
| 6164 |                                        self.cmds.push(Cmd::ToggleWorkLights); | support logic; impact depends on
| 6165 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6166 |                                    if ui.button("Road Lights").clicked() { | runtime gate; condition changes can
| 6167 |                                        self.cmds.push(Cmd::ToggleRoadLights); | support logic; impact depends on
| 6168 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6169 |                                    if ui.button("Beacon").clicked() { | runtime gate; condition changes can enable
| 6170 |                                        self.cmds.push(Cmd::ToggleBeacon); | support logic; impact depends on neigh
| 6171 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6172 |                                    if ui.button("Honk ■").clicked() { | runtime gate; condition changes can enabl
| 6173 |                                        self.cmds.push(Cmd::Honk); | support logic; impact depends on neighboring
| 6174 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6175 |                                    if ui.button("Wiper").clicked() { | runtime gate; condition changes can enable
| 6176 |                                        self.cmds.push(Cmd::CycleWiper); | support logic; impact depends on neighb
| 6177 |                                    } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6178 |                                }); | support logic; impact depends on neighboring block and data flow. |
| 6179 |                                ui.separator(); | support logic; impact depends on neighboring block and data flow
| 6180 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6181 |                                    RichText::new("BCM STATUS:") | support logic; impact depends on neighboring bl
| 6182 |                                        .size(11.0) | support logic; impact depends on neighboring block and data f
| 6183 |                                        .color(Color32::from_gray(140)), | support logic; impact depends on neighb
| 6184 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 6185 |                                let b = &self.bench.bcm; | support logic; impact depends on neighboring block and
| 6186 |                                bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 6187 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 6188 |                                    "Battery", | support logic; impact depends on neighboring block and data flow.
| 6189 |                                    b.battery_voltage - 9.0, | support logic; impact depends on neighboring block
| 6190 |                                    7.0, | support logic; impact depends on neighboring block and data flow. |
| 6191 |                                    "V", | support logic; impact depends on neighboring block and data flow. |
| 6192 |                                    110.0, | support logic; impact depends on neighboring block and data flow. |
| 6193 |                                    if b.battery_voltage < 11.5 { | runtime gate; condition changes can enable/dis
```

```
| 6194 |                                 Color32::RED | support logic; impact depends on neighboring block and data
| 6195 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 6196 |                                 Color32::GREEN | support logic; impact depends on neighboring block and da
| 6197 |                             }, | support logic; impact depends on neighboring block and data flow. |
| 6198 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 6199 |                         digital_readout( | support logic; impact depends on neighboring block and data flo
| 6200 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 6201 |                             "SoC", | support logic; impact depends on neighboring block and data flow. |
| 6202 |                             &format!("{:.0}%", b.battery_soc_pct), | support logic; impact depends on neig
| 6203 |                             Color32::from_gray(200), | support logic; impact depends on neighboring block a
| 6204 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 6205 |                         digital_readout( | support logic; impact depends on neighboring block and data flo
| 6206 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 6207 |                             "Load", | support logic; impact depends on neighboring block and data flow. |
| 6208 |                             &format!("{:.0} A", b.total_load_amps), | support logic; impact depends on nei
| 6209 |                             Color32::from_gray(180), | support logic; impact depends on neighboring block a
| 6210 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 6211 |                         digital_readout( | support logic; impact depends on neighboring block and data flo
| 6212 |                             ui, | support logic; impact depends on neighboring block and data flow. |
| 6213 |                             "Wiper", | support logic; impact depends on neighboring block and data flow. |
| 6214 |                             &format!("{:?}", b.wiper_speed), | error propagation point; changes alter faile
| 6215 |                             Color32::from_gray(180), | support logic; impact depends on neighboring block a
| 6216 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 6217 |                         for fuse in &b.fuses { | iteration logic; may alter timing, throughput, and safety
| 6218 |                             if fuse.blown { | runtime gate; condition changes can enable/disable critical |
| 6219 |                                 ui.label( | support logic; impact depends on neighboring block and data fl
| 6220 |                                     RichText::new(format!("■ FUSE BLOWN: {}", fuse.id)) | support logic; i
| 6221 |                                         .size(10.5) | support logic; impact depends on neighboring block a
| 6222 |                                         .color(Color32::RED), | support logic; impact depends on neighbori
| 6223 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 6224 |                             } | scope delimiter; structural only, but wrong placement changes ownership/li
| 6225 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6226 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6227 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 6228 |             }); | support logic; impact depends on neighboring block and data flow. |
| 6229 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6230 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6231 | (blank) | blank separator; no runtime behavior. |
| 6232 | // ■■■
```

```
| 6270 |             ), | support logic; impact depends on neighboring block and data flow. |
| 6271 |         )); | support logic; impact depends on neighboring block and data flow. |
| 6272 |         if self.uds_log.len() > 240 { | runtime gate; condition changes can enable/disable critical paths. |
| 6273 |             self.uds_log.drain(0..60); | protocol or I/O critical; changes may impact external integration cor
| 6274 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6275 |         resp | support logic; impact depends on neighboring block and data flow. |
| 6276 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6277 | (blank) | blank separator; no runtime behavior. |
| 6278 |     fn run_uds_flash_pipeline(&mut self, sa: u8, ts: f64) -> Result<String, String> { | function signature/body
| 6279 |         if self.uds_flash_path.trim().is_empty() { | runtime gate; condition changes can enable/disable critica
| 6280 |             return Err("selecione um arquivo .bin para flash".into()); | protocol or I/O critical; changes may
| 6281 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6282 |         let fw = std::fs::read(&self.uds_flash_path) | protocol or I/O critical; changes may impact external in
| 6283 |             .map_err(\|e\| format!("falha ao ler firmware: {}", e))?; | error propagation point; changes alter
| 6284 |         if fw.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 6285 |             return Err("arquivo de firmware vazio".into()); | support logic; impact depends on neighboring bloc
| 6286 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6287 | (blank) | blank separator; no runtime behavior. |
| 6288 |         // Session + security unlock for programming level. | comment/documentation; changing affects understan
| 6289 |         let _ = self.uds_send_and_log(sa, vec![0x10, 0x02], ts); | protocol or I/O critical; changes may impact
| 6290 |         let seed_resp = self.uds_send_and_log(sa, vec![0x27, 0x05], ts); | protocol or I/O critical; changes ma
| 6291 |         if seed_resp.len() < 6 \|\| seed_resp[0] != 0x67 { | runtime gate; condition changes can enable/disable
| 6292 |             return Err("seed de seguranca nao recebido".into()); | support logic; impact depends on neighboring
| 6293 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6294 |         let seed = ((seed_resp[2] as u32) << 24) | support logic; impact depends on neighboring block and data
| 6295 |             \| ((seed_resp[3] as u32) << 16) | support logic; impact depends on neighboring block and data flow
| 6296 |             \| ((seed_resp[4] as u32) << 8) | support logic; impact depends on neighboring block and data flow
| 6297 |             \| seed_resp[5] as u32; | support logic; impact depends on neighboring block and data flow. |
| 6298 |         let key = seed ^ 0xDEADBEEF; | support logic; impact depends on neighboring block and data flow. |
| 6299 |         let unlock_resp = self.uds_send_and_log( | protocol or I/O critical; changes may impact external integ
| 6300 |             sa, | support logic; impact depends on neighboring block and data flow. |
| 6301 |             vec![ | support logic; impact depends on neighboring block and data flow. |
| 6302 |                 0x27, | support logic; impact depends on neighboring block and data flow. |
| 6303 |                 0x06, | support logic; impact depends on neighboring block and data flow. |
| 6304 |                 ((key >> 24) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow
| 6305 |                 ((key >> 16) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow
| 6306 |                 ((key >> 8) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow.
| 6307 |                 (key & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 6308 |             ], | support logic; impact depends on neighboring block and data flow. |
| 6309 |             ts, | support logic; impact depends on neighboring block and data flow. |
| 6310 |         ); | support logic; impact depends on neighboring block and data flow. |
| 6311 |         if unlock_resp.first().copied() != Some(0x67) { | runtime gate; condition changes can enable/disable c
| 6312 |             return Err("unlock de seguranca falhou".into()); | support logic; impact depends on neighboring bl
| 6313 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6314 | (blank) | blank separator; no runtime behavior. |
| 6315 |         let _ = self.uds_send_and_log(sa, vec![0x10, 0x03], ts); | protocol or I/O critical; changes may impact
| 6316 | (blank) | blank separator; no runtime behavior. |
| 6317 |         let total_len = fw.len() as u32; | support logic; impact depends on neighboring block and data flow. |
| 6318 |         let req_dl = vec![ | support logic; impact depends on neighboring block and data flow. |
| 6319 |             0x34, | support logic; impact depends on neighboring block and data flow. |
| 6320 |             0x00, | support logic; impact depends on neighboring block and data flow. |
| 6321 |             0x00, | support logic; impact depends on neighboring block and data flow. |
| 6322 |             0x00, | support logic; impact depends on neighboring block and data flow. |
| 6323 |             0x00, | support logic; impact depends on neighboring block and data flow. |
| 6324 |             0x00, | support logic; impact depends on neighboring block and data flow. |
| 6325 |             ((total_len >> 24) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flo
| 6326 |             ((total_len >> 16) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flo
| 6327 |             ((total_len >> 8) & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow
| 6328 |             (total_len & 0xFF) as u8, | support logic; impact depends on neighboring block and data flow. |
| 6329 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 6330 |         let dl_resp = self.uds_send_and_log(sa, req_dl, ts); | protocol or I/O critical; changes may impact ex
| 6331 |         if dl_resp.first().copied() != Some(0x74) { | runtime gate; condition changes can enable/disable critic
| 6332 |             return Err("request download rejeitado".into()); | support logic; impact depends on neighboring bl
| 6333 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6334 | (blank) | blank separator; no runtime behavior. |
| 6335 |         let mut block: u8 = 1; | support logic; impact depends on neighboring block and data flow. |
| 6336 |         for chunk in fw.chunks(120) { | iteration logic; may alter timing, throughput, and safety constraints.
| 6337 |             let mut req = Vec::with_capacity(chunk.len() + 2); | support logic; impact depends on neighboring
| 6338 |             req.push(0x36); | support logic; impact depends on neighboring block and data flow. |
| 6339 |             req.push(block); | support logic; impact depends on neighboring block and data flow. |
| 6340 |             req.extend_from_slice(chunk); | support logic; impact depends on neighboring block and data flow.
| 6341 |             let tr = self.uds_send_and_log(sa, req, ts); | protocol or I/O critical; changes may impact externa
| 6342 |             if tr.first().copied() != Some(0x76) { | runtime gate; condition changes can enable/disable critica
| 6343 |                 return Err(format!("transfer data falhou no bloco {}", block)); | support logic; impact depends
| 6344 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6345 |             block = block.wrapping_add(1); | support logic; impact depends on neighboring block and data flow.
```

```
| 6346 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6347 | (blank) | blank separator; no runtime behavior. |
| 6348 |         let end = self.uds_send_and_log(sa, vec![0x37, 0x00], ts); | protocol or I/O critical; changes may impa
| 6349 |         if end.first().copied() != Some(0x77) { | runtime gate; condition changes can enable/disable critical p
| 6350 |             return Err("request transfer exit falhou".into()); | support logic; impact depends on neighboring b
| 6351 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6352 | (blank) | blank separator; no runtime behavior. |
| 6353 |         Ok(format!( | support logic; impact depends on neighboring block and data flow. |
| 6354 |             "flash concluido (simulado): {} bytes enviados para SA 0x{:02X}", | protocol or I/O critical; chang
| 6355 |             fw.len(), | support logic; impact depends on neighboring block and data flow. |
| 6356 |             sa | support logic; impact depends on neighboring block and data flow. |
| 6357 |         )) | support logic; impact depends on neighboring block and data flow. |
| 6358 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6359 | (blank) | blank separator; no runtime behavior. |
| 6360 |     fn leak_ascii_cad( | function signature/body; changes affect API behavior and control-flow. |
| 6361 |         circuit: &LeakCircuit, | support logic; impact depends on neighboring block and data flow. |
| 6362 |         report: &auto_breaking::CircuitResult, | support logic; impact depends on neighboring block and data f
| 6363 |     ) -> (String, Vec<String>) { | support logic; impact depends on neighboring block and data flow. |
| 6364 |         const SECTORS: usize = 16; | support logic; impact depends on neighboring block and data flow. |
| 6365 |         let mut local_pressure = [0.0_f64; SECTORS]; | support logic; impact depends on neighboring block and d
| 6366 |         let mut local_frag = [0.0_f64; SECTORS]; | support logic; impact depends on neighboring block and data
| 6367 |         let mut local_flow = [0.0_f64; SECTORS]; | support logic; impact depends on neighboring block and data
| 6368 | (blank) | blank separator; no runtime behavior. |
| 6369 |         for i in 0..SECTORS { | iteration logic; may alter timing, throughput, and safety constraints. |
| 6370 |             let theta = i as f64 * std::f64::consts::TAU / SECTORS as f64; | support logic; impact depends on n
| 6371 |             let harmonic = (2.0 * theta).cos() * 0.15 + (theta + 0.4).sin() * 0.07; | support logic; impact dep
| 6372 |             let pressure = (report.current_pressure_bar * (1.0 + harmonic)).max(0.0); | support logic; impact d
| 6373 |             let pressure_ratio = pressure / circuit.pressure.rupture_bar.max(1e-6); | support logic; impact dep
| 6374 |             let frag = (0.62 * pressure_ratio + 0.38 * (report.rupture_probability_pct / 100.0)) | support logi
| 6375 |                 .clamp(0.0, 1.0); | support logic; impact depends on neighboring block and data flow. |
| 6376 |             let flow = report.leak_lpm * (0.65 + 0.35 * harmonic.max(-0.9)); | support logic; impact depends on
| 6377 | (blank) | blank separator; no runtime behavior. |
| 6378 |             local_pressure[i] = pressure; | support logic; impact depends on neighboring block and data flow.
| 6379 |             local_frag[i] = frag; | support logic; impact depends on neighboring block and data flow. |
| 6380 |             local_flow[i] = flow.max(0.0); | support logic; impact depends on neighboring block and data flow.
| 6381 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6382 | (blank) | blank separator; no runtime behavior. |
| 6383 |         let mut max_pressure_idx = 0usize; | support logic; impact depends on neighboring block and data flow.
| 6384 |         let mut max_frag_idx = 0usize; | support logic; impact depends on neighboring block and data flow. |
| 6385 |         for i in 1..SECTORS { | iteration logic; may alter timing, throughput, and safety constraints. |
| 6386 |             if local_pressure[i] > local_pressure[max_pressure_idx] { | runtime gate; condition changes can ena
| 6387 |                 max_pressure_idx = i; | support logic; impact depends on neighboring block and data flow. |
| 6388 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6389 |             if local_frag[i] > local_frag[max_frag_idx] { | runtime gate; condition changes can enable/disable
| 6390 |                 max_frag_idx = i; | support logic; impact depends on neighboring block and data flow. |
| 6391 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6392 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6393 | (blank) | blank separator; no runtime behavior. |
| 6394 |         let width = 37usize; | support logic; impact depends on neighboring block and data flow. |
| 6395 |         let height = 17usize; | support logic; impact depends on neighboring block and data flow. |
| 6396 |         let mut grid = vec![vec![' '; width]; height]; | support logic; impact depends on neighboring block and
| 6397 |         for (y, row) in grid.iter_mut().enumerate().take(height) { | iteration logic; may alter timing, through
| 6398 |             for (x, cell) in row.iter_mut().enumerate().take(width) { | iteration logic; may alter timing, thro
| 6399 |                 let nx = (x as f64 / (width - 1) as f64 - 0.5) * 2.0; | support logic; impact depends on neighb
| 6400 |                 let ny = (y as f64 / (height - 1) as f64 - 0.5) * 2.0; | support logic; impact depends on neig
| 6401 |                 let rr = ((nx * 1.1).powi(2) + (ny * 0.9).powi(2)).sqrt(); | support logic; impact depends on n
| 6402 |                 if (0.45..=0.95).contains(&rr) { | runtime gate; condition changes can enable/disable critical
| 6403 |                     let ang = ny.atan2(nx); | support logic; impact depends on neighboring block and data flow
| 6404 |                     let mut idx = ((ang + std::f64::consts::PI) / std::f64::consts::TAU | support logic; impac
| 6405 |                         * SECTORS as f64) as usize; | support logic; impact depends on neighboring block and da
| 6406 |                     if idx >= SECTORS { | runtime gate; condition changes can enable/disable critical paths. |
| 6407 |                         idx = SECTORS - 1; | support logic; impact depends on neighboring block and data flow.
| 6408 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6409 |                     let frag = local_frag[idx]; | support logic; impact depends on neighboring block and data f
| 6410 |                     let mut ch = if frag > 0.85 { | support logic; impact depends on neighboring block and data
| 6411 |                         'X' | support logic; impact depends on neighboring block and data flow. |
| 6412 |                     } else if frag > 0.70 { | support logic; impact depends on neighboring block and data flow
| 6413 |                         '!' | support logic; impact depends on neighboring block and data flow. |
| 6414 |                     } else if frag > 0.50 { | support logic; impact depends on neighboring block and data flow
| 6415 |                         '*' | support logic; impact depends on neighboring block and data flow. |
| 6416 |                     } else if frag > 0.30 { | support logic; impact depends on neighboring block and data flow
| 6417 |                         ':' | support logic; impact depends on neighboring block and data flow. |
| 6418 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 6419 |                         '.' | support logic; impact depends on neighboring block and data flow. |
| 6420 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6421 |                     if idx == max_pressure_idx { | runtime gate; condition changes can enable/disable critical
```

```
| 6422 |                         ch = 'P'; | support logic; impact depends on neighboring block and data flow. |
| 6423 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6424 |                     if idx == max_frag_idx { | runtime gate; condition changes can enable/disable critical pat
| 6425 |                         ch = 'T'; | support logic; impact depends on neighboring block and data flow. |
| 6426 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6427 |                     *cell = ch; | support logic; impact depends on neighboring block and data flow. |
| 6428 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6429 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6430 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6431 | (blank) | blank separator; no runtime behavior. |
| 6432 |         let mut cad = String::new(); | support logic; impact depends on neighboring block and data flow. |
| 6433 |         cad.push_str("      ENGINEERING ASCII CAD - O-RING / SEAL CROSS-SECTION\n"); | support logic; impact de
| 6434 |         cad.push_str("  legend: T=tear-up point, P=max pressure point, X=fragility critical\n"); | support logi
| 6435 |         cad.push_str("          !=high fragility, *=moderate, :=watch, .=low\n\n"); | support logic; impact dep
| 6436 |         for row in &grid { | iteration logic; may alter timing, throughput, and safety constraints. |
| 6437 |             let line: String = row.iter().collect(); | support logic; impact depends on neighboring block and d
| 6438 |             cad.push_str(&line); | support logic; impact depends on neighboring block and data flow. |
| 6439 |             cad.push('\n'); | support logic; impact depends on neighboring block and data flow. |
| 6440 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6441 | (blank) | blank separator; no runtime behavior. |
| 6442 |         let mut lines = Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 6443 |         lines.push(format!( | support logic; impact depends on neighboring block and data flow. |
| 6444 |             "pressure point (max): sector {} \| {:.2} bar", | support logic; impact depends on neighboring bloc
| 6445 |             max_pressure_idx, local_pressure[max_pressure_idx] | support logic; impact depends on neighboring b
| 6446 |         )); | support logic; impact depends on neighboring block and data flow. |
| 6447 |         lines.push(format!( | support logic; impact depends on neighboring block and data flow. |
| 6448 |             "tear-up point (max fragility): sector {} \| fragility {:.3}", | support logic; impact depends on n
| 6449 |             max_frag_idx, local_frag[max_frag_idx] | support logic; impact depends on neighboring block and dat
| 6450 |         )); | support logic; impact depends on neighboring block and data flow. |
| 6451 |         lines.push(format!( | support logic; impact depends on neighboring block and data flow. |
| 6452 |             "global leak flow: {:.4} L/min \| rupture risk {:.1}% \| band {}", | support logic; impact depends
| 6453 |             report.leak_lpm, report.rupture_probability_pct, report.pressure_band | support logic; impact depen
| 6454 |         )); | support logic; impact depends on neighboring block and data flow. |
| 6455 | (blank) | blank separator; no runtime behavior. |
| 6456 |         let mut ranked: Vec<(usize, f64)> = (0..SECTORS).map(\|i\| (i, local_frag[i])).collect(); | support log
| 6457 |         ranked.sort_by(\|a, b\| b.1.total_cmp(&a.1)); | support logic; impact depends on neighboring block and
| 6458 |         for (idx, frag) in ranked.into_iter().take(6) { | iteration logic; may alter timing, throughput, and sa
| 6459 |             lines.push(format!( | support logic; impact depends on neighboring block and data flow. |
| 6460 |                 "fragility point s{:02}: pressure {:.2} bar \| fragility {:.3} \| flow {:.4} L/min", | support
| 6461 |                 idx, local_pressure[idx], frag, local_flow[idx] | support logic; impact depends on neighboring
| 6462 |             )); | support logic; impact depends on neighboring block and data flow. |
| 6463 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6464 | (blank) | blank separator; no runtime behavior. |
| 6465 |         (cad, lines) | support logic; impact depends on neighboring block and data flow. |
| 6466 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6467 | (blank) | blank separator; no runtime behavior. |
| 6468 |     fn tab_leak_lab(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and contro
| 6469 |         self.sanitize_leak_manual(); | support logic; impact depends on neighboring block and data flow. |
| 6470 |         self.sanitize_leak_custom(); | support logic; impact depends on neighboring block and data flow. |
| 6471 |         let manual_validation = self.validate_leak_manual(); | support logic; impact depends on neighboring blo
| 6472 |         let custom_validation = self.validate_leak_custom(); | support logic; impact depends on neighboring blo
| 6473 |         let manual_ok = manual_validation.is_ok(); | support logic; impact depends on neighboring block and dat
| 6474 |         let custom_ok = custom_validation.is_ok(); | support logic; impact depends on neighboring block and dat
| 6475 | (blank) | blank separator; no runtime behavior. |
| 6476 |         let circuits: Vec<(usize, String, String, String, String)> = self | support logic; impact depends on ne
| 6477 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 6478 |             .leak_rig | support logic; impact depends on neighboring block and data flow. |
| 6479 |             .circuits | support logic; impact depends on neighboring block and data flow. |
| 6480 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 6481 |             .enumerate() | support logic; impact depends on neighboring block and data flow. |
| 6482 |             .map(\|(i, c)\| { | support logic; impact depends on neighboring block and data flow. |
| 6483 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 6484 |                     i, | support logic; impact depends on neighboring block and data flow. |
| 6485 |                     c.name.clone(), | support logic; impact depends on neighboring block and data flow. |
| 6486 |                     c.application.clone(), | support logic; impact depends on neighboring block and data flow.
| 6487 |                     format!("{:?}", c.component), | error propagation point; changes alter failure propagation
| 6488 |                     format!("{:?}", c.oil_type), | error propagation point; changes alter failure propagation s
| 6489 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 6490 |             }) | support logic; impact depends on neighboring block and data flow. |
| 6491 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 6492 |         let reports = self.bench.leak_reports.clone(); | support logic; impact depends on neighboring block and
| 6493 |         let alerts = self.bench.leak_rig.alerts.clone(); | support logic; impact depends on neighboring block a
| 6494 |         let predictions = self.leak_predictions.clone(); | support logic; impact depends on neighboring block a
| 6495 |         let calibration_report = self.leak_calibration_report.clone(); | support logic; impact depends on neigh
| 6496 |         let selected_circuit = self.bench.leak_rig.circuits.get(self.leak_sel_idx).cloned(); | support logic; i
| 6497 |         let selected_report = selected_circuit.as_ref().and_then(\|c\| { | support logic; impact depends on nei
```

```
| 6498 |             reports | support logic; impact depends on neighboring block and data flow. |
| 6499 |                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 6500 |                 .find(\|r\| r.name == c.name) | support logic; impact depends on neighboring block and data flo
| 6501 |                 .cloned() | support logic; impact depends on neighboring block and data flow. |
| 6502 |         }); | support logic; impact depends on neighboring block and data flow. |
| 6503 | (blank) | blank separator; no runtime behavior. |
| 6504 |         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 6505 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6506 |                 RichText::new("LEAK PHYSICS LAB — industrial manual + automatic workflow") | support logic; imp
| 6507 |                     .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 6508 |                     .color(Color32::from_rgb(80, 155, 255)) | support logic; impact depends on neighboring bloc
| 6509 |                     .strong(), | support logic; impact depends on neighboring block and data flow. |
| 6510 |             ); | support logic; impact depends on neighboring block and data flow. |
| 6511 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6512 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6513 |                 RichText::new(format!("{} circuits", circuits.len())) | support logic; impact depends on neighb
| 6514 |                     .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 6515 |                     .color(Color32::from_gray(150)), | support logic; impact depends on neighboring block and d
| 6516 |             ); | support logic; impact depends on neighboring block and data flow. |
| 6517 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6518 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6519 |                 RichText::new(format!( | support logic; impact depends on neighboring block and data flow. |
| 6520 |                     "Total leak {:.3} L/min", | support logic; impact depends on neighboring block and data flo
| 6521 |                     self.bench.leak_rig.total_leak_lpm | support logic; impact depends on neighboring block and
| 6522 |                 )) | support logic; impact depends on neighboring block and data flow. |
| 6523 |                 .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 6524 |                 .color(Color32::YELLOW), | support logic; impact depends on neighboring block and data flow. |
| 6525 |             ); | support logic; impact depends on neighboring block and data flow. |
| 6526 |             if !self.leak_note.is_empty() { | runtime gate; condition changes can enable/disable critical paths
| 6527 |                 ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6528 |                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6529 |                     RichText::new(&self.leak_note) | support logic; impact depends on neighboring block and da
| 6530 |                         .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 6531 |                         .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighboring block and d
| 6532 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 6533 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6534 |         }); | support logic; impact depends on neighboring block and data flow. |
| 6535 |         ui.group(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 6536 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6537 |                 RichText::new("Como ler esta aba (direto ao ponto)") | support logic; impact depends on neighbo
| 6538 |                     .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 6539 |                     .color(Color32::from_rgb(110, 180, 255)) | support logic; impact depends on neighboring blo
| 6540 |                     .strong(), | support logic; impact depends on neighboring block and data flow. |
| 6541 |             ); | support logic; impact depends on neighboring block and data flow. |
| 6542 |             ui.label("1) Runtime Circuits mostra estado atual (pressao, leak, risco, alerta)."); | support logi
| 6543 |             ui.label("2) Manual Input ajusta engenharia (envelope/oring/oil). A aplicacao invalida fica bloquea
| 6544 |             ui.label("3) Scenario Ranking ordena pior caso por risco e pico de pressao."); | support logic; imp
| 6545 |             ui.label("4) Timeback ASCII/Plot mostra evolucao temporal de pressao-risco-vazao."); | support logi
| 6546 |             ui.label("5) Use export CSV/JSON para auditoria e rastreabilidade."); | support logic; impact depen
| 6547 |         }); | support logic; impact depends on neighboring block and data flow. |
| 6548 |         ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 6549 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6550 |                 .add(Button::new("Nova rodada (limpar saidas)").fill(Color32::from_rgb(40, 50, 70))) | support
| 6551 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 6552 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6553 |                 self.cmds.push(Cmd::LeakClearScenarioOutputs); | support logic; impact depends on neighboring b
| 6554 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6555 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6556 |                 .add(Button::new("RESET Leak Sim (estado fisico)").fill(Color32::from_rgb(85, 35, 25))) | suppo
| 6557 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 6558 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6559 |                 self.cmds.push(Cmd::LeakResetSimulation); | support logic; impact depends on neighboring block
| 6560 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6561 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6562 |                 .add_enabled( | support logic; impact depends on neighboring block and data flow. |
| 6563 |                     manual_ok, | support logic; impact depends on neighboring block and data flow. |
| 6564 |                     Button::new("Aplicar + Rodar Cenario").fill(Color32::from_rgb(20, 70, 100)), | support logi
| 6565 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 6566 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 6567 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6568 |                 self.cmds.push(Cmd::LeakApplyAndPredict); | support logic; impact depends on neighboring block
| 6569 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6570 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 6571 |                 .add_enabled( | support logic; impact depends on neighboring block and data flow. |
| 6572 |                     manual_ok, | support logic; impact depends on neighboring block and data flow. |
| 6573 |                     Button::new("Aplicar + Rodar Monte Carlo").fill(Color32::from_rgb(70, 35, 90)), | support
```

```
| 6574 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 6575 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 6576 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6577 |                 self.cmds.push(Cmd::LeakApplyAndMonteCarlo); | support logic; impact depends on neighboring blo
| 6578 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6579 |         }); | support logic; impact depends on neighboring block and data flow. |
| 6580 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6581 |         ui.group(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 6582 |             ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6583 |                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6584 |                     RichText::new("TIMEBACK VIEW") | support logic; impact depends on neighboring block and da
| 6585 |                         .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 6586 |                         .color(Color32::from_rgb(110, 180, 255)) | support logic; impact depends on neighboring
| 6587 |                         .strong(), | support logic; impact depends on neighboring block and data flow. |
| 6588 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 6589 |                 ui.label("janela(s)"); | support logic; impact depends on neighboring block and data flow. |
| 6590 |                 ui.add(DragValue::new(&mut self.leak_timeback_window_s).speed(5.0).range(10.0..=7200.0)); | su
| 6591 |                 ui.label("stride"); | support logic; impact depends on neighboring block and data flow. |
| 6592 |                 ui.add(DragValue::new(&mut self.leak_timeback_stride).speed(1.0).range(1..=60)); | support log
| 6593 |                 ui.checkbox(&mut self.leak_timeback_latest_first, "latest first"); | support logic; impact dep
| 6594 |                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6595 |                     RichText::new(format!("hist: {} pontos", self.leak_temporal_trace.len())) | support logic;
| 6596 |                         .size(9.8) | support logic; impact depends on neighboring block and data flow. |
| 6597 |                         .color(Color32::from_gray(160)), | support logic; impact depends on neighboring block a
| 6598 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 6599 |             }); | support logic; impact depends on neighboring block and data flow. |
| 6600 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6601 |                 RichText::new("ASCII timeline ponto a ponto: [pressao][risco][vazao] com janela temporal confi
| 6602 |                     .size(9.6) | support logic; impact depends on neighboring block and data flow. |
| 6603 |                     .color(Color32::from_gray(155)), | support logic; impact depends on neighboring block and
| 6604 |             ); | support logic; impact depends on neighboring block and data flow. |
| 6605 |         }); | support logic; impact depends on neighboring block and data flow. |
| 6606 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6607 | (blank) | blank separator; no runtime behavior. |
| 6608 |         let mut do_apply = false; | support logic; impact depends on neighboring block and data flow. |
| 6609 |         let mut do_predict = false; | support logic; impact depends on neighboring block and data flow. |
| 6610 |         let mut do_add_custom = false; | support logic; impact depends on neighboring block and data flow. |
| 6611 |         let mut do_export_report_csv = false; | support logic; impact depends on neighboring block and data fl
| 6612 |         let mut do_export_report_json = false; | support logic; impact depends on neighboring block and data f
| 6613 |         let mut do_export_pred_csv = false; | support logic; impact depends on neighboring block and data flow
| 6614 |         let mut do_export_pred_json = false; | support logic; impact depends on neighboring block and data flo
| 6615 |         let mut do_export_catalog_mat_csv = false; | support logic; impact depends on neighboring block and da
| 6616 |         let mut do_export_catalog_mat_json = false; | support logic; impact depends on neighboring block and d
| 6617 |         let mut do_export_catalog_oil_csv = false; | support logic; impact depends on neighboring block and da
| 6618 |         let mut do_export_catalog_oil_json = false; | support logic; impact depends on neighboring block and d
| 6619 |         let mut do_pick_calibration_csv = false; | support logic; impact depends on neighboring block and data
| 6620 |         let mut do_run_calibration = false; | support logic; impact depends on neighboring block and data flow
| 6621 |         let mut do_export_calibration_csv = false; | support logic; impact depends on neighboring block and da
| 6622 |         let mut do_export_calibration_json = false; | support logic; impact depends on neighboring block and d
| 6623 |         let mut do_monte_carlo = false; | support logic; impact depends on neighboring block and data flow. |
| 6624 |         let mut pending_select: Option<usize> = None; | support logic; impact depends on neighboring block and
| 6625 | (blank) | blank separator; no runtime behavior. |
| 6626 |         ScrollArea::vertical().id_salt("leak::outer_scroll").auto_shrink([false, false]).max_height(ui.availab
| 6627 |             ui.columns(3, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 6628 |                 // Left: circuit list + runtime status | comment/documentation; changing affects understanding
| 6629 |                 egui::Frame::group(cols[0].style()).fill(Color32::from_gray(15)).show(&mut cols[0], \|ui\| { |
| 6630 |                     ui.label(RichText::new("RUNTIME CIRCUITS").size(11.5).color(Color32::from_rgb(80,155,255))
| 6631 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6632 |                     for (i, name, app, comp, oil) in &circuits { | iteration logic; may alter timing, throughpu
| 6633 |                         ui.push_id(format!("leak::circuit_row_{}", i), \|ui\| { | support logic; impact depends
| 6634 |                             let sel = *i == self.leak_sel_idx; | support logic; impact depends on neighboring b
| 6635 |                             let fill = if sel { Color32::from_rgb(30,60,100) } else { Color32::from_gray(25) }
| 6636 |                             if ui.add(Button::new(RichText::new(format!("{} [{} / {}]", name, comp, oil)).size
| 6637 |                                 pending_select = Some(*i); | support logic; impact depends on neighboring bloc
| 6638 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6639 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 6640 |                         ui.label(RichText::new(app).size(9.8).color(Color32::from_gray(155))); | support logic
| 6641 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6642 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6643 |                     ui.label(RichText::new("REAL-TIME REPORT").size(11.0).color(Color32::from_rgb(80,155,255))
| 6644 |                     for r in &reports { | iteration logic; may alter timing, throughput, and safety constraints
| 6645 |                         let c = match r.alert { | support logic; impact depends on neighboring block and data f
| 6646 |                             auto_breaking::LeakAlertLevel::Normal => Color32::GREEN, | support logic; impact de
| 6647 |                             auto_breaking::LeakAlertLevel::Watch => Color32::YELLOW, | support logic; impact de
| 6648 |                             auto_breaking::LeakAlertLevel::Warning => Color32::from_rgb(255,170,60), | support
| 6649 |                             auto_breaking::LeakAlertLevel::Critical => Color32::RED, | support logic; impact de
```

```
| 6650 |                             auto_breaking::LeakAlertLevel::Ruptured => Color32::from_rgb(255,0,0), | support lo
| 6651 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 6652 |                         ui.label(RichText::new(format!( | support logic; impact depends on neighboring block an
| 6653 |                             "{} \| {} \| p={:.1}bar ({}) \| leak {:.3} L/min \| risk {:.0}%", | support logic;
| 6654 |                             r.name, | support logic; impact depends on neighboring block and data flow. |
| 6655 |                             r.alert, | support logic; impact depends on neighboring block and data flow. |
| 6656 |                             r.current_pressure_bar, | support logic; impact depends on neighboring block and da
| 6657 |                             r.pressure_band, | support logic; impact depends on neighboring block and data flow
| 6658 |                             r.leak_lpm, | support logic; impact depends on neighboring block and data flow. |
| 6659 |                             r.rupture_probability_pct | support logic; impact depends on neighboring block and
| 6660 |                         )).size(10.0).color(c)); | support logic; impact depends on neighboring block and data
| 6661 |                         ui.label(RichText::new(format!( | support logic; impact depends on neighboring block an
| 6662 |                             "safe {:.1}-{:.1}bar target {:.1}bar \| rupture at {} bar / {} h", | support logic
| 6663 |                             r.recommended_hold_min_bar, | support logic; impact depends on neighboring block an
| 6664 |                             r.recommended_hold_max_bar, | support logic; impact depends on neighboring block an
| 6665 |                             r.recommended_hold_target_bar, | support logic; impact depends on neighboring bloc
| 6666 |                             r.rupture_pressure_bar.map(\|v\| format!("{v:.2}")).unwrap_or_else(\|\| "n/a".into
| 6667 |                             r.rupture_elapsed_h.map(\|v\| format!("{v:.3}")).unwrap_or_else(\|\| "n/a".into())
| 6668 |                         )).size(9.5).color(Color32::from_gray(160))); | support logic; impact depends on neigh
| 6669 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6670 |                     if !alerts.is_empty() { | runtime gate; condition changes can enable/disable critical paths
| 6671 |                         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6672 |                         ui.label(RichText::new("ACTIVE ALERTS").size(11.0).color(Color32::RED)); | support log
| 6673 |                         for a in &alerts { | iteration logic; may alter timing, throughput, and safety constra
| 6674 |                             ui.label(RichText::new(format!("{}: {}", a.circuit_name, a.message)).size(10.0).co
| 6675 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 6676 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6677 | (blank) | blank separator; no runtime behavior. |
| 6678 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6679 |                     ui.label(RichText::new("3D PHYSICS/LEAK VIEW").size(11.0).color(Color32::from_rgb(80,155,25
| 6680 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6681 |                         ui.label("Yaw"); | support logic; impact depends on neighboring block and data flow. |
| 6682 |                         ui.add(Slider::new(&mut self.leak_view_yaw_deg, -180.0..=180.0).show_value(true)); | su
| 6683 |                         ui.label("Pitch"); | support logic; impact depends on neighboring block and data flow.
| 6684 |                         ui.add(Slider::new(&mut self.leak_view_pitch_deg, -80.0..=80.0).show_value(true)); | su
| 6685 |                         ui.label("Zoom"); | support logic; impact depends on neighboring block and data flow.
| 6686 |                         ui.add(Slider::new(&mut self.leak_view_zoom, 0.4..=2.5).show_value(true)); | support lo
| 6687 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6688 |                     let (rect, _resp) = ui.allocate_exact_size( | support logic; impact depends on neighboring
| 6689 |                         Vec2::new(ui.available_width() - 4.0, 180.0), | support logic; impact depends on neigh
| 6690 |                         Sense::hover(), | support logic; impact depends on neighboring block and data flow. |
| 6691 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 6692 |                     let p = ui.painter_at(rect); | support logic; impact depends on neighboring block and data
| 6693 |                     p.rect_filled(rect, 4.0, Color32::from_gray(10)); | support logic; impact depends on neigh
| 6694 |                     p.rect_stroke(rect, 4.0, Stroke::new(1.0, Color32::from_gray(60))); | support logic; impac
| 6695 | (blank) | blank separator; no runtime behavior. |
| 6696 |                     let yaw = self.leak_view_yaw_deg.to_radians(); | support logic; impact depends on neighbor
| 6697 |                     let pitch = self.leak_view_pitch_deg.to_radians(); | support logic; impact depends on neig
| 6698 |                     let zoom = self.leak_view_zoom; | support logic; impact depends on neighboring block and da
| 6699 |                     let center = rect.center(); | support logic; impact depends on neighboring block and data
| 6700 |                     let scale = 70.0 * zoom; | support logic; impact depends on neighboring block and data flow
| 6701 | (blank) | blank separator; no runtime behavior. |
| 6702 |                     let project = \|x: f32, y: f32, z: f32\| -> Pos2 { | support logic; impact depends on neig
| 6703 |                         let cy = yaw.cos(); | support logic; impact depends on neighboring block and data flow
| 6704 |                         let sy = yaw.sin(); | support logic; impact depends on neighboring block and data flow
| 6705 |                         let cp = pitch.cos(); | support logic; impact depends on neighboring block and data flo
| 6706 |                         let sp = pitch.sin(); | support logic; impact depends on neighboring block and data flo
| 6707 | (blank) | blank separator; no runtime behavior. |
| 6708 |                         let xr = x * cy + z * sy; | support logic; impact depends on neighboring block and data
| 6709 |                         let zr = -x * sy + z * cy; | support logic; impact depends on neighboring block and da
| 6710 |                         let yr = y * cp - zr * sp; | support logic; impact depends on neighboring block and da
| 6711 |                         let zr2 = y * sp + zr * cp + 8.0; | support logic; impact depends on neighboring block
| 6712 |                         let f = scale / zr2.max(0.3); | support logic; impact depends on neighboring block and
| 6713 |                         Pos2::new(center.x + xr * f, center.y - yr * f) | support logic; impact depends on neig
| 6714 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6715 | (blank) | blank separator; no runtime behavior. |
| 6716 |                     let axis = [ | support logic; impact depends on neighboring block and data flow. |
| 6717 |                         ([-2.2, 0.0, 0.0], [2.2, 0.0, 0.0], Color32::from_rgb(220, 90, 90)), | support logic;
| 6718 |                         ([0.0, 0.0, -2.2], [0.0, 0.0, 2.2], Color32::from_rgb(90, 220, 90)), | support logic;
| 6719 |                         ([0.0, 0.0, 0.0], [0.0, 2.2, 0.0], Color32::from_rgb(90, 140, 255)), | support logic;
| 6720 |                     ]; | support logic; impact depends on neighboring block and data flow. |
| 6721 |                     for (a, b, c) in axis { | iteration logic; may alter timing, throughput, and safety constra
| 6722 |                         p.line_segment( | support logic; impact depends on neighboring block and data flow. |
| 6723 |                             [project(a[0], a[1], a[2]), project(b[0], b[1], b[2])], | support logic; impact dep
| 6724 |                             Stroke::new(1.2, c), | support logic; impact depends on neighboring block and data
| 6725 |                         ); | support logic; impact depends on neighboring block and data flow. |
```

```
| 6726 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6727 | (blank) | blank separator; no runtime behavior. |
| 6728 |                     for (i, r) in reports.iter().take(8).enumerate() { | iteration logic; may alter timing, thr
| 6729 |                         let x = -1.8 + i as f32 * 0.5; | support logic; impact depends on neighboring block and
| 6730 |                         let h = (r.leak_lpm as f32 * 0.9).clamp(0.08, 1.8); | support logic; impact depends on
| 6731 |                         let risk = (r.rupture_probability_pct as f32 / 100.0).clamp(0.0, 1.0); | support logic
| 6732 |                         let col = Color32::from_rgb( | support logic; impact depends on neighboring block and
| 6733 |                             (90.0 + 150.0 * risk) as u8, | support logic; impact depends on neighboring block
| 6734 |                             (210.0 - 120.0 * risk) as u8, | support logic; impact depends on neighboring block
| 6735 |                             90, | support logic; impact depends on neighboring block and data flow. |
| 6736 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 6737 | (blank) | blank separator; no runtime behavior. |
| 6738 |                         let y0 = 0.0f32; | support logic; impact depends on neighboring block and data flow. |
| 6739 |                         let y1 = h; | support logic; impact depends on neighboring block and data flow. |
| 6740 |                         let z0 = -0.15f32; | support logic; impact depends on neighboring block and data flow.
| 6741 |                         let z1 = 0.15f32; | support logic; impact depends on neighboring block and data flow.
| 6742 |                         let x0 = x; | support logic; impact depends on neighboring block and data flow. |
| 6743 |                         let x1 = x + 0.3; | support logic; impact depends on neighboring block and data flow.
| 6744 | (blank) | blank separator; no runtime behavior. |
| 6745 |                         let corners = [ | support logic; impact depends on neighboring block and data flow. |
| 6746 |                             project(x0, y0, z0), | support logic; impact depends on neighboring block and data
| 6747 |                             project(x1, y0, z0), | support logic; impact depends on neighboring block and data
| 6748 |                             project(x1, y1, z0), | support logic; impact depends on neighboring block and data
| 6749 |                             project(x0, y1, z0), | support logic; impact depends on neighboring block and data
| 6750 |                             project(x0, y0, z1), | support logic; impact depends on neighboring block and data
| 6751 |                             project(x1, y0, z1), | support logic; impact depends on neighboring block and data
| 6752 |                             project(x1, y1, z1), | support logic; impact depends on neighboring block and data
| 6753 |                             project(x0, y1, z1), | support logic; impact depends on neighboring block and data
| 6754 |                         ]; | support logic; impact depends on neighboring block and data flow. |
| 6755 | (blank) | blank separator; no runtime behavior. |
| 6756 |                         let edges = [ | support logic; impact depends on neighboring block and data flow. |
| 6757 |                             (0, 1), (1, 2), (2, 3), (3, 0), | support logic; impact depends on neighboring bloc
| 6758 |                             (4, 5), (5, 6), (6, 7), (7, 4), | support logic; impact depends on neighboring bloc
| 6759 |                             (0, 4), (1, 5), (2, 6), (3, 7), | support logic; impact depends on neighboring bloc
| 6760 |                         ]; | support logic; impact depends on neighboring block and data flow. |
| 6761 |                         for (a, b) in edges { | iteration logic; may alter timing, throughput, and safety cons
| 6762 |                             p.line_segment([corners[a], corners[b]], Stroke::new(1.0, col)); | support logic;
| 6763 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 6764 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6765 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 6766 | (blank) | blank separator; no runtime behavior. |
| 6767 |                 // Middle: manual params | comment/documentation; changing affects understanding and intent tra
| 6768 |                 egui::Frame::group(cols[1].style()).fill(Color32::from_gray(15)).show(&mut cols[1], \|ui\| { |
| 6769 |                     ui.label(RichText::new("MANUAL ENGINEERING INPUT").size(11.5).color(Color32::from_rgb(80,15
| 6770 |                     ui.label(RichText::new("Use this for production-calibrated pressure/oring/oil settings").si
| 6771 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6772 | (blank) | blank separator; no runtime behavior. |
| 6773 |                     let oil_types = OilType::all(); | support logic; impact depends on neighboring block and da
| 6774 |                     ComboBox::from_id_salt("leak::manual_oil_type") | support logic; impact depends on neighbor
| 6775 |                         .width(180.0) | support logic; impact depends on neighboring block and data flow. |
| 6776 |                         .selected_text( | support logic; impact depends on neighboring block and data flow. |
| 6777 |                             oil_types | support logic; impact depends on neighboring block and data flow. |
| 6778 |                                 .get(self.leak_manual.oil_type_idx) | support logic; impact depends on neighbor
| 6779 |                                 .copied() | support logic; impact depends on neighboring block and data flow.
| 6780 |                                 .unwrap_or(OilType::Custom) | support logic; impact depends on neighboring bloc
| 6781 |                                 .name(), | support logic; impact depends on neighboring block and data flow. |
| 6782 |                         ) | support logic; impact depends on neighboring block and data flow. |
| 6783 |                         .show_ui(ui, \|ui\| { | support logic; impact depends on neighboring block and data flo
| 6784 |                             for (i, o) in oil_types.iter().enumerate() { | iteration logic; may alter timing,
| 6785 |                                 ui.selectable_value(&mut self.leak_manual.oil_type_idx, i, o.name()); | support
| 6786 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 6787 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 6788 |                     ui.label(RichText::new("Oil Type").size(9.6).color(Color32::from_gray(150))); | support log
| 6789 | (blank) | blank separator; no runtime behavior. |
| 6790 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6791 |                         ui.label("Piston bar"); | support logic; impact depends on neighboring block and data
| 6792 |                         ui.add(DragValue::new(&mut self.leak_manual.piston_pressure_bar).speed(0.2).range(0.0.
| 6793 |                         ui.label("Op bar"); | support logic; impact depends on neighboring block and data flow
| 6794 |                         ui.add(DragValue::new(&mut self.leak_manual.operation_pressure_bar).speed(0.2).range(0
| 6795 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6796 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6797 |                         ui.label("Min"); ui.add(DragValue::new(&mut self.leak_manual.pressure_min_bar).speed(0
| 6798 |                         ui.label("Mean"); ui.add(DragValue::new(&mut self.leak_manual.pressure_mean_bar).speed
| 6799 |                         ui.label("Ideal"); ui.add(DragValue::new(&mut self.leak_manual.pressure_ideal_bar).spe
| 6800 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6801 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
```

```
| 6802 |                         ui.label("Max"); ui.add(DragValue::new(&mut self.leak_manual.pressure_max_bar).speed(0
| 6803 |                         ui.label("Rupture"); ui.add(DragValue::new(&mut self.leak_manual.pressure_rupture_bar)
| 6804 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6805 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6806 |                         ui.label("Squeeze %"); ui.add(DragValue::new(&mut self.leak_manual.squeeze_pct).speed(
| 6807 |                         ui.label("Comp set %"); ui.add(DragValue::new(&mut self.leak_manual.compression_set_pc
| 6808 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6809 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6810 |                         ui.label("Base leak mm²"); | support logic; impact depends on neighboring block and da
| 6811 |                         ui.add(DragValue::new(&mut self.leak_manual.base_leak_area_mm2).speed(0.0001).range(0.
| 6812 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6813 | (blank) | blank separator; no runtime behavior. |
| 6814 |                     if let Err(msg) = &manual_validation { | runtime gate; condition changes can enable/disabl
| 6815 |                         ui.label(RichText::new(format!("■ {}", msg)).size(9.8).color(Color32::YELLOW)); | supp
| 6816 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6817 | (blank) | blank separator; no runtime behavior. |
| 6818 |                     if ui.add_enabled( | runtime gate; condition changes can enable/disable critical paths. |
| 6819 |                         manual_ok, | support logic; impact depends on neighboring block and data flow. |
| 6820 |                         Button::new(RichText::new("APPLY MANUAL PARAMETERS").size(11.0)).fill(Color32::from_rg
| 6821 |                     ).clicked() { | support logic; impact depends on neighboring block and data flow. |
| 6822 |                         do_apply = true; | support logic; impact depends on neighboring block and data flow. |
| 6823 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6824 | (blank) | blank separator; no runtime behavior. |
| 6825 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6826 |                     ui.label(RichText::new("AUTO PREDICTION").size(11.0).color(Color32::from_rgb(80,155,255)))
| 6827 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6828 |                         ui.label("Horizon s"); | support logic; impact depends on neighboring block and data f
| 6829 |                         ui.add(DragValue::new(&mut self.leak_horizon_s).speed(5.0).range(60.0..=86400.0)); | s
| 6830 |                         ui.label("dt s"); | support logic; impact depends on neighboring block and data flow.
| 6831 |                         ui.add(DragValue::new(&mut self.leak_scenario_dt).speed(0.01).range(0.01..=1.0)); | su
| 6832 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 6833 |                     if ui.add(Button::new(RichText::new("RUN SCENARIO PREDICTION").size(11.0)).fill(Color32::f
| 6834 |                         do_predict = true; | support logic; impact depends on neighboring block and data flow.
| 6835 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6836 |                     if ui.add(Button::new(RichText::new("RUN MONTE CARLO (120 runs)").size(11.0)).fill(Color32
| 6837 |                         do_monte_carlo = true; | support logic; impact depends on neighboring block and data f
| 6838 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 6839 | (blank) | blank separator; no runtime behavior. |
| 6840 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6841 |                     ui.label(RichText::new("O-RING / SEAL ENGINEERING RENDER (ASCII CAD)").size(11.0).color(Co
| 6842 |                     egui::CollapsingHeader::new("ASCII legend guide") | support logic; impact depends on neigh
| 6843 |                         .default_open(false) | support logic; impact depends on neighboring block and data flo
| 6844 |                         .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 6845 |                             ui.label(RichText::new("CAD symbols:").size(9.4).color(Color32::from_gray(180)));
| 6846 |                             ui.label(RichText::new("P = ponto de maior pressao local").size(9.2).color(Color32
| 6847 |                             ui.label(RichText::new("T = ponto de maior fragilidade (tear-up)").size(9.2).color
| 6848 |                             ui.label(RichText::new("X/!/*/:/. = criticidade decrescente da fragilidade").size(
| 6849 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 6850 |                             ui.label(RichText::new("Timeline symbols:").size(9.4).color(Color32::from_gray(180
| 6851 |                             ui.label(RichText::new("1o char = pressao: ^ alto, : medio, . baixo").size(9.2).co
| 6852 |                             ui.label(RichText::new("2o char = risco: ! alto, * medio, . baixo").size(9.2).colo
| 6853 |                             ui.label(RichText::new("3o char = vazao: ~ alta, - media, . baixa").size(9.2).colo
| 6854 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 6855 |                     if let (Some(c), Some(r)) = (&selected_circuit, &selected_report) { | runtime gate; condit
| 6856 |                         let (cad, points) = Self::leak_ascii_cad(c, r); | support logic; impact depends on nei
| 6857 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6858 |                             RichText::new("Use scroll horizontal para ver o desenho ASCII completo sem corte."
| 6859 |                                 .size(9.2) | support logic; impact depends on neighboring block and data flow.
| 6860 |                                 .color(Color32::from_gray(150)), | support logic; impact depends on neighborin
| 6861 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 6862 |                         ScrollArea::both() | support logic; impact depends on neighboring block and data flow.
| 6863 |                             .id_salt("leak::ascii_cad_scroll") | support logic; impact depends on neighboring
| 6864 |                             .max_height(245.0) | support logic; impact depends on neighboring block and data f
| 6865 |                             .auto_shrink([false, false]) | support logic; impact depends on neighboring block
| 6866 |                             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data f
| 6867 |                                 ui.set_min_width(360.0); | support logic; impact depends on neighboring block
| 6868 |                                 ui.label(RichText::new(cad).size(8.8).monospace().color(Color32::from_gray(190
| 6869 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 6870 |                         for p in points { | iteration logic; may alter timing, throughput, and safety constrai
| 6871 |                             ui.label(RichText::new(p).size(9.2).color(Color32::from_gray(170))); | support log
| 6872 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 6873 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6874 |                             RichText::new(format!( | support logic; impact depends on neighboring block and da
| 6875 |                                 "{} \| material {} \| cs {:.2}mm \| squeeze {:.1}% \| gap {:.3}mm \| comp set
| 6876 |                                 c.name, | support logic; impact depends on neighboring block and data flow. |
| 6877 |                                 c.spec.material.name(), | support logic; impact depends on neighboring block a
```

```
| 6878 |                                c.spec.cross_section_mm, | support logic; impact depends on neighboring block a
| 6879 |                                c.spec.squeeze_pct, | support logic; impact depends on neighboring block and da
| 6880 |                                c.spec.extrusion_gap_mm, | support logic; impact depends on neighboring block a
| 6881 |                                c.spec.compression_set_pct, | support logic; impact depends on neighboring bloc
| 6882 |                                c.spec.shore_a, | support logic; impact depends on neighboring block and data f
| 6883 |                                c.spec.design_life_hours | support logic; impact depends on neighboring block a
| 6884 |                            )) | support logic; impact depends on neighboring block and data flow. |
| 6885 |                            .size(9.4) | support logic; impact depends on neighboring block and data flow. |
| 6886 |                            .color(Color32::from_gray(170)), | support logic; impact depends on neighboring blo
| 6887 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 6888 | (blank) | blank separator; no runtime behavior. |
| 6889 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 6890 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6891 |                            RichText::new("TEMPORAL ASCII EVOLUTION (pressure/risk/flow)") | support logic; imp
| 6892 |                                .size(10.6) | support logic; impact depends on neighboring block and data flow
| 6893 |                                .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on nei
| 6894 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 6895 |                        let t_now = self.bench.elapsed; | support logic; impact depends on neighboring block ar
| 6896 |                        let t_min = (t_now - self.leak_timeback_window_s).max(0.0); | support logic; impact dep
| 6897 |                        let stride = self.leak_timeback_stride.max(1); | support logic; impact depends on neigh
| 6898 |                        let mut tb: Vec<(f64, f64, f64, f64)> = self | support logic; impact depends on neighbo
| 6899 |                            .leak_temporal_trace | support logic; impact depends on neighboring block and data
| 6900 |                            .iter() | support logic; impact depends on neighboring block and data flow. |
| 6901 |                            .copied() | support logic; impact depends on neighboring block and data flow. |
| 6902 |                            .filter(\|(t, _, _, _)\| *t >= t_min) | support logic; impact depends on neighborin
| 6903 |                            .step_by(stride) | support logic; impact depends on neighboring block and data flow
| 6904 |                            .collect(); | support logic; impact depends on neighboring block and data flow. |
| 6905 |                        if self.leak_timeback_latest_first { | runtime gate; condition changes can enable/disal
| 6906 |                            tb.reverse(); | support logic; impact depends on neighboring block and data flow.
| 6907 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 6908 | (blank) | blank separator; no runtime behavior. |
| 6909 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6910 |                            RichText::new(format!( | support logic; impact depends on neighboring block and dat
| 6911 |                                "window {:.0}s \| samples {} \| order {}", | support logic; impact depends on n
| 6912 |                                self.leak_timeback_window_s, | support logic; impact depends on neighboring blo
| 6913 |                                tb.len(), | support logic; impact depends on neighboring block and data flow.
| 6914 |                                if self.leak_timeback_latest_first { | runtime gate; condition changes can enal
| 6915 |                                    "new->old" | support logic; impact depends on neighboring block and data f
| 6916 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 6917 |                                    "old->new" | support logic; impact depends on neighboring block and data f
| 6918 |                                } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 6919 |                            )) | support logic; impact depends on neighboring block and data flow. |
| 6920 |                            .size(9.6) | support logic; impact depends on neighboring block and data flow. |
| 6921 |                            .color(Color32::from_gray(160)), | support logic; impact depends on neighboring blo
| 6922 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 6923 | (blank) | blank separator; no runtime behavior. |
| 6924 |                        let mut timeline = String::new(); | support logic; impact depends on neighboring block
| 6925 |                        timeline.push_str("legend [P][R][F]: P(^ high, : mid, . low) \| R(! high, * mid, . low
| 6926 |                        timeline.push_str("idx \| time(s) \| ascii\n"); | support logic; impact depends on neig
| 6927 |                        for (i, (tt, p, risk, leak)) in tb.iter().enumerate() { | iteration logic; may alter ti
| 6928 |                            let p_ratio = (p / c.pressure.rupture_bar.max(1e-6)).clamp(0.0, 1.0); | support log
| 6929 |                            let r_ratio = (risk / 100.0).clamp(0.0, 1.0); | support logic; impact depends on ne
| 6930 |                            let f_ratio = (leak / 1.5).clamp(0.0, 1.0); | support logic; impact depends on neig
| 6931 |                            let pch = if p_ratio > 0.85 { | support logic; impact depends on neighboring block
| 6932 |                                '^' | support logic; impact depends on neighboring block and data flow. |
| 6933 |                            } else if p_ratio > 0.55 { | support logic; impact depends on neighboring block and
| 6934 |                                ':' | support logic; impact depends on neighboring block and data flow. |
| 6935 |                            } else { | support logic; impact depends on neighboring block and data flow. |
| 6936 |                                '.' | support logic; impact depends on neighboring block and data flow. |
| 6937 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 6938 |                            let rch = if r_ratio > 0.80 { | support logic; impact depends on neighboring block
| 6939 |                                '!' | support logic; impact depends on neighboring block and data flow. |
| 6940 |                            } else if r_ratio > 0.45 { | support logic; impact depends on neighboring block and
| 6941 |                                '*' | support logic; impact depends on neighboring block and data flow. |
| 6942 |                            } else { | support logic; impact depends on neighboring block and data flow. |
| 6943 |                                '.' | support logic; impact depends on neighboring block and data flow. |
| 6944 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 6945 |                            let fch = if f_ratio > 0.80 { | support logic; impact depends on neighboring block
| 6946 |                                '~' | support logic; impact depends on neighboring block and data flow. |
| 6947 |                            } else if f_ratio > 0.45 { | support logic; impact depends on neighboring block and
| 6948 |                                '-' | support logic; impact depends on neighboring block and data flow. |
| 6949 |                            } else { | support logic; impact depends on neighboring block and data flow. |
| 6950 |                                '.' | support logic; impact depends on neighboring block and data flow. |
| 6951 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 6952 |                            timeline.push_str(&format!("{:03} \| {:7.2} \| {}{}{}\n", i, tt, pch, rch, fch));
| 6953 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
```

```
| 6954 |                         ScrollArea::both() | support logic; impact depends on neighboring block and data flow.
| 6955 |                             .id_salt("leak::timeback_ascii_scroll") | support logic; impact depends on neighbor
| 6956 |                             .max_height(155.0) | support logic; impact depends on neighboring block and data fl
| 6957 |                             .auto_shrink([false, false]) | support logic; impact depends on neighboring block a
| 6958 |                             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data fl
| 6959 |                                 ui.set_min_width(340.0); | support logic; impact depends on neighboring block a
| 6960 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 6961 |                                     RichText::new(timeline) | support logic; impact depends on neighboring bloc
| 6962 |                                         .size(8.9) | support logic; impact depends on neighboring block and dat
| 6963 |                                         .monospace() | support logic; impact depends on neighboring block and c
| 6964 |                                         .color(Color32::from_gray(178)), | support logic; impact depends on nei
| 6965 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 6966 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 6967 | (blank) | blank separator; no runtime behavior. |
| 6968 |                         let p_points: Vec<[f64; 2]> = tb.iter().map(\|(t, p, _, _)\| [*t, *p]).collect(); | sup
| 6969 |                         let r_points: Vec<[f64; 2]> = tb.iter().map(\|(t, _, r, _)\| [*t, *r]).collect(); | sup
| 6970 |                         let f_points: Vec<[f64; 2]> = tb.iter().map(\|(t, _, _, f)\| [*t, *f]).collect(); | sup
| 6971 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 6972 |                             RichText::new("TIMEBACK VISUAL AID (P/R/F)") | support logic; impact depends on nei
| 6973 |                                 .size(10.4) | support logic; impact depends on neighboring block and data flow
| 6974 |                                 .color(Color32::from_rgb(110, 180, 255)), | support logic; impact depends on ne
| 6975 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 6976 |                         Plot::new("leak::timeback_plot") | support logic; impact depends on neighboring block a
| 6977 |                             .height(150.0) | support logic; impact depends on neighboring block and data flow.
| 6978 |                             .allow_drag(false) | support logic; impact depends on neighboring block and data fl
| 6979 |                             .allow_zoom(false) | support logic; impact depends on neighboring block and data fl
| 6980 |                             .allow_scroll(false) | support logic; impact depends on neighboring block and data
| 6981 |                             .show_axes([true, true]) | support logic; impact depends on neighboring block and c
| 6982 |                             .show(ui, \|plot_ui\| { | support logic; impact depends on neighboring block and da
| 6983 |                                 let lp = PlotPoints::from_iter(p_points.iter().copied()); | support logic; impa
| 6984 |                                 let lr = PlotPoints::from_iter(r_points.iter().copied()); | support logic; impa
| 6985 |                                 let lf = PlotPoints::from_iter(f_points.iter().copied()); | support logic; impa
| 6986 |                                 plot_ui.line( | support logic; impact depends on neighboring block and data flo
| 6987 |                                     Line::new(lp) | support logic; impact depends on neighboring block and data
| 6988 |                                         .color(Color32::from_rgb(255, 170, 60)) | support logic; impact depends
| 6989 |                                         .name("Pressure bar"), | support logic; impact depends on neighboring b
| 6990 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 6991 |                                 plot_ui.line( | support logic; impact depends on neighboring block and data flo
| 6992 |                                     Line::new(lr) | support logic; impact depends on neighboring block and data
| 6993 |                                         .color(Color32::from_rgb(255, 80, 80)) | support logic; impact depends
| 6994 |                                         .name("Risk %"), | support logic; impact depends on neighboring block a
| 6995 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 6996 |                                 plot_ui.line( | support logic; impact depends on neighboring block and data flo
| 6997 |                                     Line::new(lf) | support logic; impact depends on neighboring block and data
| 6998 |                                         .color(Color32::from_rgb(90, 220, 180)) | support logic; impact depends
| 6999 |                                         .name("Leak LPM"), | support logic; impact depends on neighboring block
| 7000 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 7001 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 7002 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 7003 |                         ui.label(RichText::new("Select a circuit with runtime data to render O-ring CAD").size
| 7004 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7005 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 7006 | (blank) | blank separator; no runtime behavior. |
| 7007 |                 // Right: custom circuit + scenario table | comment/documentation; changing affects understandi
| 7008 |                 egui::Frame::group(cols[2].style()).fill(Color32::from_gray(15)).show(&mut cols[2], \|ui\| { |
| 7009 |                     ui.label(RichText::new("CUSTOM CIRCUIT BUILDER").size(11.5).color(Color32::from_rgb(80,155,
| 7010 |                     ui.push_id("leak::custom_name_row", \|ui\| { | support logic; impact depends on neighboring
| 7011 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7012 |                         ui.label("Name"); | support logic; impact depends on neighboring block and data flow.
| 7013 |                         ui.add_sized([120.0, 20.0], TextEdit::singleline(&mut self.leak_custom.name)); | suppo
| 7014 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7015 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7016 |                     ui.push_id("leak::custom_app_row", \|ui\| { | support logic; impact depends on neighboring
| 7017 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7018 |                         ui.label("App"); | support logic; impact depends on neighboring block and data flow. |
| 7019 |                         ui.add_sized([180.0, 20.0], TextEdit::singleline(&mut self.leak_custom.application));
| 7020 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7021 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7022 |                     const COMP_NAMES: [&str; 3] = ["O-ring", "Seal", "A/C Hose"]; | support logic; impact depen
| 7023 |                     let oil_types = OilType::all(); | support logic; impact depends on neighboring block and da
| 7024 |                     let materials = OringMaterial::all(); | support logic; impact depends on neighboring block
| 7025 |                     ComboBox::from_id_salt("leak::custom_component").selected_text(COMP_NAMES[self.leak_custom
| 7026 |                         for (i, n) in COMP_NAMES.iter().enumerate() { ui.selectable_value(&mut self.leak_custo
| 7027 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7028 |                     ComboBox::from_id_salt("leak::custom_oil").selected_text( | support logic; impact depends o
| 7029 |                         oil_types | support logic; impact depends on neighboring block and data flow. |
```

```
| 7030 |                             .get(self.leak_custom.oil_type_idx) | support logic; impact depends on neighboring
| 7031 |                             .copied() | support logic; impact depends on neighboring block and data flow. |
| 7032 |                             .unwrap_or(OilType::Custom) | support logic; impact depends on neighboring block a
| 7033 |                             .name(), | support logic; impact depends on neighboring block and data flow. |
| 7034 |                         ).show_ui(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7035 |                             for (i, o) in oil_types.iter().enumerate() { ui.selectable_value(&mut self.leak_custom
| 7036 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7037 |                         ComboBox::from_id_salt("leak::custom_material").selected_text( | support logic; impact depe
| 7038 |                             materials | support logic; impact depends on neighboring block and data flow. |
| 7039 |                             .get(self.leak_custom.material_idx) | support logic; impact depends on neighboring
| 7040 |                             .copied() | support logic; impact depends on neighboring block and data flow. |
| 7041 |                             .unwrap_or(OringMaterial::Nbr) | support logic; impact depends on neighboring bloc
| 7042 |                             .name(), | support logic; impact depends on neighboring block and data flow. |
| 7043 |                         ).show_ui(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7044 |                             for (i, m) in materials.iter().enumerate() { ui.selectable_value(&mut self.leak_custom
| 7045 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7046 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7047 |                             ui.label("Seals"); ui.add(DragValue::new(&mut self.leak_custom.seal_count).range(1..=1
| 7048 |                             ui.label("ShoreA"); ui.add(DragValue::new(&mut self.leak_custom.shore_a).range(40.0..=9
| 7049 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7050 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7051 |                             ui.label("Piston"); ui.add(DragValue::new(&mut self.leak_custom.piston_pressure_bar).sp
| 7052 |                             ui.label("Op"); ui.add(DragValue::new(&mut self.leak_custom.operation_pressure_bar).spe
| 7053 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7054 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7055 |                             ui.label("Min"); ui.add(DragValue::new(&mut self.leak_custom.min_bar).speed(0.2)); | su
| 7056 |                             ui.label("Mean"); ui.add(DragValue::new(&mut self.leak_custom.mean_bar).speed(0.2)); |
| 7057 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7058 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7059 |                             ui.label("Ideal"); ui.add(DragValue::new(&mut self.leak_custom.ideal_bar).speed(0.2));
| 7060 |                             ui.label("Max"); ui.add(DragValue::new(&mut self.leak_custom.max_bar).speed(0.2)); | su
| 7061 |                             ui.label("Rupt"); ui.add(DragValue::new(&mut self.leak_custom.rupture_bar).speed(0.2))
| 7062 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7063 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7064 |                             ui.label("cs mm"); ui.add(DragValue::new(&mut self.leak_custom.cross_section_mm).speed
| 7065 |                             ui.label("sq %"); ui.add(DragValue::new(&mut self.leak_custom.squeeze_pct).speed(0.1))
| 7066 |                             ui.label("gap"); ui.add(DragValue::new(&mut self.leak_custom.extrusion_gap_mm).speed(0
| 7067 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7068 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7069 |                             ui.label("comp set"); ui.add(DragValue::new(&mut self.leak_custom.compression_set_pct)
| 7070 |                             ui.label("life h"); ui.add(DragValue::new(&mut self.leak_custom.design_life_hours).spe
| 7071 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7072 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7073 |                             ui.label("base leak"); ui.add(DragValue::new(&mut self.leak_custom.base_leak_area_mm2)
| 7074 |                             ui.label("Cd"); ui.add(DragValue::new(&mut self.leak_custom.discharge_coeff).speed(0.01
| 7075 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7076 |                         ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7077 |                             ui.label("Reservoir L"); ui.add(DragValue::new(&mut self.leak_custom.reservoir_volume_
| 7078 |                             ui.label("Support LPM"); ui.add(DragValue::new(&mut self.leak_custom.support_lpm).speed
| 7079 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7080 | (blank) | blank separator; no runtime behavior. |
| 7081 |                         if let Err(msg) = &custom_validation { | runtime gate; condition changes can enable/disable
| 7082 |                             ui.label(RichText::new(format!("■ {}", msg)).size(9.8).color(Color32::YELLOW)); | supp
| 7083 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7084 | (blank) | blank separator; no runtime behavior. |
| 7085 |                         if ui.add_enabled( | runtime gate; condition changes can enable/disable critical paths. |
| 7086 |                             custom_ok, | support logic; impact depends on neighboring block and data flow. |
| 7087 |                             Button::new(RichText::new("ADD CUSTOM CIRCUIT").size(11.0)).fill(Color32::from_rgb(60,4
| 7088 |                         ).clicked() { | support logic; impact depends on neighboring block and data flow. |
| 7089 |                             do_add_custom = true; | support logic; impact depends on neighboring block and data flo
| 7090 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7091 | (blank) | blank separator; no runtime behavior. |
| 7092 |                         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 7093 |                         ui.label(RichText::new("SCENARIO RANKING").size(11.0).color(Color32::from_rgb(80,155,255))
| 7094 |                         ScrollArea::vertical().id_salt("leak::scenario_ranking_scroll").max_height(190.0).auto_shri
| 7095 |                             for p in predictions.iter().take(50) { | iteration logic; may alter timing, throughput,
| 7096 |                                 let c = match p.final_alert { | support logic; impact depends on neighboring block
| 7097 |                                     auto_breaking::LeakAlertLevel::Normal => Color32::GREEN, | support logic; impac
| 7098 |                                     auto_breaking::LeakAlertLevel::Watch => Color32::YELLOW, | support logic; impac
| 7099 |                                     auto_breaking::LeakAlertLevel::Warning => Color32::from_rgb(255,170,60), | supp
| 7100 |                                     auto_breaking::LeakAlertLevel::Critical => Color32::RED, | support logic; impac
| 7101 |                                     auto_breaking::LeakAlertLevel::Ruptured => Color32::from_rgb(255,0,0), | suppo
| 7102 |                                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 7103 |                                 ui.label(RichText::new(format!("{} \| {} \| {} \| risk {:.0}% \| pmax {:.1}bar", p
| 7104 |                                 ui.label(RichText::new(format!("mode: {} \| ttf: {}", p.likely_failure_mode, p.hour
| 7105 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
```

```
| 7106 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7107 |                     ui.label(RichText::new("3D SCENARIO SIMULATION VIEW").size(10.6).color(Color32::from_rgb(80
| 7108 |                     let (srect, _sresp) = ui.allocate_exact_size( | support logic; impact depends on neighborin
| 7109 |                         Vec2::new(ui.available_width() - 4.0, 170.0), | support logic; impact depends on neighb
| 7110 |                         Sense::hover(), | support logic; impact depends on neighboring block and data flow. |
| 7111 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 7112 |                     let sp = ui.painter_at(srect); | support logic; impact depends on neighboring block and dat
| 7113 |                     sp.rect_filled(srect, 4.0, Color32::from_gray(10)); | support logic; impact depends on neig
| 7114 |                     sp.rect_stroke(srect, 4.0, Stroke::new(1.0, Color32::from_gray(60))); | support logic; impa
| 7115 | (blank) | blank separator; no runtime behavior. |
| 7116 |                     let center = srect.center(); | support logic; impact depends on neighboring block and data
| 7117 |                     let yaw = 28.0f32.to_radians(); | support logic; impact depends on neighboring block and da
| 7118 |                     let pitch = 20.0f32.to_radians(); | support logic; impact depends on neighboring block and
| 7119 |                     let scale = 65.0f32; | support logic; impact depends on neighboring block and data flow. |
| 7120 |                     let project = \|x: f32, y: f32, z: f32\| -> Pos2 { | support logic; impact depends on neigh
| 7121 |                         let cy = yaw.cos(); | support logic; impact depends on neighboring block and data flow
| 7122 |                         let sy = yaw.sin(); | support logic; impact depends on neighboring block and data flow
| 7123 |                         let cp = pitch.cos(); | support logic; impact depends on neighboring block and data flo
| 7124 |                         let spv = pitch.sin(); | support logic; impact depends on neighboring block and data f
| 7125 | (blank) | blank separator; no runtime behavior. |
| 7126 |                         let xr = x * cy + z * sy; | support logic; impact depends on neighboring block and data
| 7127 |                         let zr = -x * sy + z * cy; | support logic; impact depends on neighboring block and dat
| 7128 |                         let yr = y * cp - zr * spv; | support logic; impact depends on neighboring block and da
| 7129 |                         let zr2 = y * spv + zr * cp + 8.5; | support logic; impact depends on neighboring bloc
| 7130 |                         let f = scale / zr2.max(0.35); | support logic; impact depends on neighboring block and
| 7131 |                         Pos2::new(center.x + xr * f, center.y - yr * f) | support logic; impact depends on neig
| 7132 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7133 | (blank) | blank separator; no runtime behavior. |
| 7134 |                     let axis = [ | support logic; impact depends on neighboring block and data flow. |
| 7135 |                         ([-2.4, 0.0, 0.0], [2.4, 0.0, 0.0], Color32::from_rgb(220, 90, 90)), | support logic;
| 7136 |                         ([0.0, 0.0, -2.4], [0.0, 0.0, 2.4], Color32::from_rgb(90, 220, 90)), | support logic;
| 7137 |                         ([0.0, 0.0, 0.0], [0.0, 2.4, 0.0], Color32::from_rgb(90, 140, 255)), | support logic;
| 7138 |                     ]; | support logic; impact depends on neighboring block and data flow. |
| 7139 |                     for (a, b, c) in axis { | iteration logic; may alter timing, throughput, and safety constra
| 7140 |                         sp.line_segment( | support logic; impact depends on neighboring block and data flow. |
| 7141 |                             [project(a[0], a[1], a[2]), project(b[0], b[1], b[2])], | support logic; impact dep
| 7142 |                             Stroke::new(1.1, c), | support logic; impact depends on neighboring block and data
| 7143 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 7144 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7145 | (blank) | blank separator; no runtime behavior. |
| 7146 |                     for (i, scen) in predictions.iter().take(36).enumerate() { | iteration logic; may alter ti
| 7147 |                         let x = -2.0 + (i % 12) as f32 * 0.36; | support logic; impact depends on neighboring
| 7148 |                         let z = -1.6 + (i / 12) as f32 * 1.0; | support logic; impact depends on neighboring b
| 7149 |                         let risk = (scen.final_rupture_probability_pct as f32 / 100.0).clamp(0.0, 1.0); | suppo
| 7150 |                         let peak = (scen.peak_pressure_bar as f32 / 320.0).clamp(0.0, 1.0); | support logic; i
| 7151 |                         let h = (0.15 + 1.9 * (0.6 * risk + 0.4 * peak)).clamp(0.1, 2.2); | support logic; impa
| 7152 |                         let col = Color32::from_rgb( | support logic; impact depends on neighboring block and
| 7153 |                             (80.0 + 170.0 * risk) as u8, | support logic; impact depends on neighboring block
| 7154 |                             (210.0 - 140.0 * risk) as u8, | support logic; impact depends on neighboring block
| 7155 |                             (90.0 + 90.0 * (1.0 - peak)) as u8, | support logic; impact depends on neighboring
| 7156 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 7157 | (blank) | blank separator; no runtime behavior. |
| 7158 |                         let p0 = project(x, 0.0, z); | support logic; impact depends on neighboring block and
| 7159 |                         let p1 = project(x, h, z); | support logic; impact depends on neighboring block and dat
| 7160 |                         sp.line_segment([p0, p1], Stroke::new(2.0, col)); | support logic; impact depends on ne
| 7161 |                         sp.circle_filled(p1, 2.5, col); | support logic; impact depends on neighboring block a
| 7162 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7163 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7164 |                         RichText::new("3D axes: X=scenario index, Y=risk/pressure amplitude, Z=scenario group"
| 7165 |                             .size(9.0) | support logic; impact depends on neighboring block and data flow. |
| 7166 |                             .color(Color32::from_gray(145)), | support logic; impact depends on neighboring blo
| 7167 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 7168 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 7169 |                     ui.label(RichText::new("EXPORT REPORTS").size(11.0).color(Color32::from_rgb(80,155,255)));
| 7170 |                     ui.push_id("leak::export_reports_row", \|ui\| { | support logic; impact depends on neighbo
| 7171 |                     ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and dat
| 7172 |                         if ui.button("CSV Runtime").clicked() { do_export_report_csv = true; } | runtime gate;
| 7173 |                         if ui.button("JSON Runtime").clicked() { do_export_report_json = true; } | runtime gate
| 7174 |                         if ui.button("CSV Prediction").clicked() { do_export_pred_csv = true; } | runtime gate
| 7175 |                         if ui.button("JSON Prediction").clicked() { do_export_pred_json = true; } | runtime ga
| 7176 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7177 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7178 |                     ui.label(RichText::new("ENGINEERING CATALOG EXPORT").size(11.0).color(Color32::from_rgb(80,
| 7179 |                     ui.push_id("leak::export_catalog_row", \|ui\| { | support logic; impact depends on neighbo
| 7180 |                     ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and dat
| 7181 |                         if ui.button("CSV Materials").clicked() { do_export_catalog_mat_csv = true; } | runtime
```

```
| 7182 |                         if ui.button("JSON Materials").clicked() { do_export_catalog_mat_json = true; } | runti
| 7183 |                         if ui.button("CSV Oils").clicked() { do_export_catalog_oil_csv = true; } | runtime gate
| 7184 |                         if ui.button("JSON Oils").clicked() { do_export_catalog_oil_json = true; } | runtime ga
| 7185 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7186 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7187 | (blank) | blank separator; no runtime behavior. |
| 7188 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 7189 |                     ui.label(RichText::new("CALIBRATION MODE (BENCH CSV)").size(11.0).color(Color32::from_rgb(8
| 7190 |                     ui.push_id("leak::calib_controls", \|ui\| { | support logic; impact depends on neighboring
| 7191 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 7192 |                         if ui.button("Select CSV").clicked() { do_pick_calibration_csv = true; } | runtime gate
| 7193 |                         let mut run_cal = ui.add_enabled(!self.leak_calibration_csv_path.is_empty(), Button::ne
| 7194 |                         if self.leak_calibration_csv_path.is_empty() { | runtime gate; condition changes can en
| 7195 |                             run_cal = run_cal.on_disabled_hover_text("Selecione um CSV de bancada antes de cali
| 7196 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 7197 |                         if run_cal.clicked() { | runtime gate; condition changes can enable/disable critical pa
| 7198 |                             do_run_calibration = true; | support logic; impact depends on neighboring block and
| 7199 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 7200 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7201 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7202 |                     if self.leak_calibration_csv_path.is_empty() { | runtime gate; condition changes can enable
| 7203 |                         ui.label(RichText::new("No CSV selected").size(9.8).color(Color32::from_gray(145))); |
| 7204 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 7205 |                         ui.label(RichText::new(format!("CSV: {}", self.leak_calibration_csv_path)).size(9.6).co
| 7206 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7207 |                     if let Some(rep) = &calibration_report { | runtime gate; condition changes can enable/disal
| 7208 |                         ui.label(RichText::new(format!( | support logic; impact depends on neighboring block a
| 7209 |                             "Calibrated circuits: {} \| samples: {}", | support logic; impact depends on neigh
| 7210 |                             rep.calibrated_circuits, rep.total_samples | support logic; impact depends on neig
| 7211 |                         )).size(9.8).color(Color32::YELLOW)); | support logic; impact depends on neighboring b
| 7212 |                         for r in rep.circuit_reports.iter().take(5) { | iteration logic; may alter timing, thre
| 7213 |                             ui.label(RichText::new(format!( | support logic; impact depends on neighboring bloc
| 7214 |                                 "{} \| RMSE {:.4} LPM \| MAPE {:.2}% \| rupture acc {:.1}%", | support logic; i
| 7215 |                                 r.circuit_name, r.rmse_leak_lpm, r.mape_leak_pct, r.rupture_accuracy_pct | supp
| 7216 |                             )).size(9.4).color(Color32::from_gray(165))); | support logic; impact depends on ne
| 7217 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 7218 |                         ui.push_id("leak::export_calibration_row", \|ui\| { | support logic; impact depends on
| 7219 |                         ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and
| 7220 |                             if ui.button("CSV Calibration Report").clicked() { do_export_calibration_csv = true
| 7221 |                             if ui.button("JSON Calibration Report").clicked() { do_export_calibration_json = tr
| 7222 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7223 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7224 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7225 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 7226 |             }); | support logic; impact depends on neighboring block and data flow. |
| 7227 |         }); | support logic; impact depends on neighboring block and data flow. |
| 7228 | (blank) | blank separator; no runtime behavior. |
| 7229 |         if let Some(i) = pending_select { | runtime gate; condition changes can enable/disable critical paths.
| 7230 |             self.cmds.push(Cmd::LeakSelectCircuit(i)); | support logic; impact depends on neighboring block and
| 7231 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7232 |         if do_apply && manual_ok { | runtime gate; condition changes can enable/disable critical paths. |
| 7233 |             self.cmds.push(Cmd::LeakApplyManual); | support logic; impact depends on neighboring block and data
| 7234 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7235 |         if do_predict { | runtime gate; condition changes can enable/disable critical paths. |
| 7236 |             self.cmds.push(Cmd::LeakPredictScenarios); | support logic; impact depends on neighboring block and
| 7237 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7238 |         if do_add_custom && custom_ok { | runtime gate; condition changes can enable/disable critical paths. |
| 7239 |             self.cmds.push(Cmd::LeakAddCustomCircuit); | support logic; impact depends on neighboring block and
| 7240 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7241 |         if do_export_report_csv { | runtime gate; condition changes can enable/disable critical paths. |
| 7242 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring block
| 7243 |             let name = format!("leak_report_{}.csv", ts); | support logic; impact depends on neighboring block
| 7244 |             if let Some(path) = Self::pick_save_path(&name, "csv") { | runtime gate; condition changes can enab
| 7245 |                 self.cmds.push(Cmd::LeakExportReportCsv(path)); | support logic; impact depends on neighboring
| 7246 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7247 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7248 |         if do_export_report_json { | runtime gate; condition changes can enable/disable critical paths. |
| 7249 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring block
| 7250 |             let name = format!("leak_report_{}.json", ts); | support logic; impact depends on neighboring block
| 7251 |             if let Some(path) = Self::pick_save_path(&name, "json") { | runtime gate; condition changes can ena
| 7252 |                 self.cmds.push(Cmd::LeakExportReportJson(path)); | support logic; impact depends on neighboring
| 7253 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7254 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7255 |         if do_export_pred_csv { | runtime gate; condition changes can enable/disable critical paths. |
| 7256 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring block
| 7257 |             let name = format!("leak_predictions_{}.csv", ts); | support logic; impact depends on neighboring b
```

```
| 7258 |             if let Some(path) = Self::pick_save_path(&name, "csv") { | runtime gate; condition changes can enal
| 7259 |                 self.cmds.push(Cmd::LeakExportPredCsv(path)); | support logic; impact depends on neighboring bl
| 7260 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7261 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7262 |         if do_export_pred_json { | runtime gate; condition changes can enable/disable critical paths. |
| 7263 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring bloc
| 7264 |             let name = format!("leak_predictions_{}.json", ts); | support logic; impact depends on neighboring
| 7265 |             if let Some(path) = Self::pick_save_path(&name, "json") { | runtime gate; condition changes can ena
| 7266 |                 self.cmds.push(Cmd::LeakExportPredJson(path)); | support logic; impact depends on neighboring b
| 7267 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7268 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7269 |         if do_monte_carlo { | runtime gate; condition changes can enable/disable critical paths. |
| 7270 |             self.cmds.push(Cmd::LeakRunMonteCarlo); | support logic; impact depends on neighboring block and da
| 7271 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7272 |         if do_export_catalog_mat_csv { | runtime gate; condition changes can enable/disable critical paths. |
| 7273 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring bloc
| 7274 |             let name = format!("materials_catalog_{}.csv", ts); | support logic; impact depends on neighboring
| 7275 |             if let Some(path) = Self::pick_save_path(&name, "csv") { | runtime gate; condition changes can enal
| 7276 |                 self.cmds.push(Cmd::LeakExportCatalogMaterialsCsv(path)); | support logic; impact depends on ne
| 7277 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7278 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7279 |         if do_export_catalog_mat_json { | runtime gate; condition changes can enable/disable critical paths. |
| 7280 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring bloc
| 7281 |             let name = format!("materials_catalog_{}.json", ts); | support logic; impact depends on neighborin
| 7282 |             if let Some(path) = Self::pick_save_path(&name, "json") { | runtime gate; condition changes can ena
| 7283 |                 self.cmds.push(Cmd::LeakExportCatalogMaterialsJson(path)); | support logic; impact depends on n
| 7284 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7285 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7286 |         if do_export_catalog_oil_csv { | runtime gate; condition changes can enable/disable critical paths. |
| 7287 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring bloc
| 7288 |             let name = format!("oils_catalog_{}.csv", ts); | support logic; impact depends on neighboring bloc
| 7289 |             if let Some(path) = Self::pick_save_path(&name, "csv") { | runtime gate; condition changes can enal
| 7290 |                 self.cmds.push(Cmd::LeakExportCatalogOilsCsv(path)); | support logic; impact depends on neighbo
| 7291 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7292 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7293 |         if do_export_catalog_oil_json { | runtime gate; condition changes can enable/disable critical paths. |
| 7294 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring bloc
| 7295 |             let name = format!("oils_catalog_{}.json", ts); | support logic; impact depends on neighboring bloc
| 7296 |             if let Some(path) = Self::pick_save_path(&name, "json") { | runtime gate; condition changes can ena
| 7297 |                 self.cmds.push(Cmd::LeakExportCatalogOilsJson(path)); | support logic; impact depends on neighb
| 7298 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7299 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7300 |         if do_pick_calibration_csv { | runtime gate; condition changes can enable/disable critical paths. |
| 7301 |             if let Some(path) = Self::pick_open_path("csv") { | runtime gate; condition changes can enable/disa
| 7302 |                 self.leak_calibration_csv_path = path; | support logic; impact depends on neighboring block and
| 7303 |                 self.leak_note = "CSV de bancada selecionado".into(); | support logic; impact depends on neighb
| 7304 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7305 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7306 |         if do_run_calibration { | runtime gate; condition changes can enable/disable critical paths. |
| 7307 |             self.cmds.push(Cmd::LeakCalibrateFromCsv( | support logic; impact depends on neighboring block and
| 7308 |                 self.leak_calibration_csv_path.clone(), | support logic; impact depends on neighboring block a
| 7309 |             )); | support logic; impact depends on neighboring block and data flow. |
| 7310 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7311 |         if do_export_calibration_csv { | runtime gate; condition changes can enable/disable critical paths. |
| 7312 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring bloc
| 7313 |             let name = format!("leak_calibration_report_{}.csv", ts); | support logic; impact depends on neighb
| 7314 |             if let Some(path) = Self::pick_save_path(&name, "csv") { | runtime gate; condition changes can enal
| 7315 |                 self.cmds.push(Cmd::LeakExportCalibrationCsv(path)); | support logic; impact depends on neighbo
| 7316 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7317 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7318 |         if do_export_calibration_json { | runtime gate; condition changes can enable/disable critical paths. |
| 7319 |             let ts = Local::now().format("%Y%m%d_%H%M%S"); | support logic; impact depends on neighboring bloc
| 7320 |             let name = format!("leak_calibration_report_{}.json", ts); | support logic; impact depends on neig
| 7321 |             if let Some(path) = Self::pick_save_path(&name, "json") { | runtime gate; condition changes can ena
| 7322 |                 self.cmds.push(Cmd::LeakExportCalibrationJson(path)); | support logic; impact depends on neighb
| 7323 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7324 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7325 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7326 | (blank) | blank separator; no runtime behavior. |
| 7327 |     fn tab_electrical(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and con
| 7328 |         let bcm = &self.bench.bcm; | support logic; impact depends on neighboring block and data flow. |
| 7329 |         let vcm = &self.bench.vcm; | support logic; impact depends on neighboring block and data flow. |
| 7330 |         let electrical = self.bench.electrical.clone(); | support logic; impact depends on neighboring block a
| 7331 |         let snapshot = electrical.snapshot.clone(); | support logic; impact depends on neighboring block and da
| 7332 |         let branches = electrical.branches.clone(); | support logic; impact depends on neighboring block and da
| 7333 |         let ecm_on = self.electrical_node_online(auto_breaking::j1939::addr::ECM_1); | support logic; impact de
```

```
| 7334 |         let tcm_on = self.electrical_node_online(auto_breaking::j1939::addr::TRANSMISSION); | support logic; im
| 7335 |         let abs_on = self.electrical_node_online(auto_breaking::j1939::addr::BRAKES); | support logic; impact d
| 7336 |         let hcm_on = self.electrical_node_online(auto_breaking::j1939::addr::HITCH); | support logic; impact de
| 7337 |         let bcm_on = self.electrical_node_online(auto_breaking::j1939::addr::CAB); | support logic; impact depe
| 7338 |         let icm_on = self.electrical_node_online(auto_breaking::j1939::addr::INSTRUMENT); | support logic; impa
| 7339 |         let body_load_a = branches | support logic; impact depends on neighboring block and data flow. |
| 7340 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 7341 |             .find(\|b\| b.id == "BODY-LOAD") | support logic; impact depends on neighboring block and data flow
| 7342 |             .map(\|b\| b.actual_current_a) | support logic; impact depends on neighboring block and data flow.
| 7343 |             .unwrap_or(0.0); | support logic; impact depends on neighboring block and data flow. |
| 7344 |         let battery_low = bcm.battery_voltage < 11.8; | support logic; impact depends on neighboring block and
| 7345 |         let any_blown = bcm.fuses.iter().any(\|f\| f.blown) \|\| branches.iter().any(\|b\| b.blown); | support
| 7346 | (blank) | blank separator; no runtime behavior. |
| 7347 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 7348 |             .id_salt("electrical::page_scroll") | support logic; impact depends on neighboring block and data
| 7349 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 7350 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 7351 |                 ui.group(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 7352 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7353 |                         RichText::new("ELECTRICAL SCHEME — live topology + harness baseline") | support logic;
| 7354 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 7355 |                             .color(Color32::from_rgb(110, 180, 255)) | support logic; impact depends on neighbo
| 7356 |                             .strong(), | support logic; impact depends on neighboring block and data flow. |
| 7357 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 7358 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7359 |                         RichText::new( | support logic; impact depends on neighboring block and data flow. |
| 7360 |                             "Live values come from BCM/VCM/boot state. Wire gauge and capacitance are engineer
| 7361 |                         ) | support logic; impact depends on neighboring block and data flow. |
| 7362 |                         .size(9.8) | support logic; impact depends on neighboring block and data flow. |
| 7363 |                         .color(Color32::from_gray(170)), | support logic; impact depends on neighboring block a
| 7364 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 7365 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 7366 | (blank) | blank separator; no runtime behavior. |
| 7367 |                 ui.add_space(6.0); | support logic; impact depends on neighboring block and data flow. |
| 7368 |                 ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data f
| 7369 |                     if ui.button("Work Lights").clicked() { | runtime gate; condition changes can enable/disab
| 7370 |                         self.cmds.push(Cmd::ToggleWorkLights); | support logic; impact depends on neighboring
| 7371 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7372 |                     if ui.button("Road Lights").clicked() { | runtime gate; condition changes can enable/disab
| 7373 |                         self.cmds.push(Cmd::ToggleRoadLights); | support logic; impact depends on neighboring
| 7374 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7375 |                     if ui.button("Beacon").clicked() { | runtime gate; condition changes can enable/disable cri
| 7376 |                         self.cmds.push(Cmd::ToggleBeacon); | support logic; impact depends on neighboring bloc
| 7377 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7378 |                     if ui.button("HVAC").clicked() { | runtime gate; condition changes can enable/disable crit
| 7379 |                         self.cmds.push(Cmd::ToggleHvac); | support logic; impact depends on neighboring block a
| 7380 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7381 |                     if ui.button("Horn").clicked() { | runtime gate; condition changes can enable/disable crit
| 7382 |                         self.cmds.push(Cmd::Honk); | support logic; impact depends on neighboring block and da
| 7383 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7384 |                     if ui.button("Wiper").clicked() { | runtime gate; condition changes can enable/disable cri
| 7385 |                         self.cmds.push(Cmd::CycleWiper); | support logic; impact depends on neighboring block a
| 7386 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7387 |                     if ui | runtime gate; condition changes can enable/disable critical paths. |
| 7388 |                         .add_enabled(any_blown, Button::new("Reset all fuses")) | support logic; impact depends
| 7389 |                         .clicked() | support logic; impact depends on neighboring block and data flow. |
| 7390 |                     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7391 |                         self.cmds.push(Cmd::ResetBcmFuses); | support logic; impact depends on neighboring bloc
| 7392 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7393 |                     if ui.button("Reset electrical faults").clicked() { | runtime gate; condition changes can
| 7394 |                         self.cmds.push(Cmd::ElectricalResetFaults); | support logic; impact depends on neighbo
| 7395 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7396 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 7397 | (blank) | blank separator; no runtime behavior. |
| 7398 |                 ui.add_space(6.0); | support logic; impact depends on neighboring block and data flow. |
| 7399 |                 egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data f
| 7400 |                     .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data
| 7401 |                     .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 7402 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7403 |                             RichText::new("30 / 15 / 31 + RELAY / GROUND CONTROL") | support logic; impact depe
| 7404 |                                 .size(11.0) | support logic; impact depends on neighboring block and data flow
| 7405 |                                 .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on ne
| 7406 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 7407 |                         ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block an
| 7408 |                             ui.label(RichText::new(format!("Line30 {:.2}V", snapshot.line30_v)).color(Color32:
| 7409 |                             ui.label(RichText::new(format!("Line15-ACC {:.2}V", snapshot.line15_acc_v)).color(
```

```
| 7410 |                             ui.label(RichText::new(format!("Line15-IGN {:.2}V", snapshot.line15_ign_v)).color(
| 7411 |                             ui.label(RichText::new(format!("Line31 drop {:.2}V", snapshot.ground_drop_v)).colo
| 7412 |                             ui.label(RichText::new(format!("Total I {:.1}A", snapshot.total_current_a)).color(
| 7413 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 7414 |                         for (point, title, fault) in [ | iteration logic; may alter timing, throughput, and sa
| 7415 |                             (ElectricalControlPoint::MainGround, "Main ground", electrical.main_ground_fault),
| 7416 |                             (ElectricalControlPoint::AccessoryRelay, "Accessory relay", electrical.accessory_r
| 7417 |                             (ElectricalControlPoint::IgnitionRelay, "Ignition relay", electrical.ignition_rela
| 7418 |                         ] { | support logic; impact depends on neighboring block and data flow. |
| 7419 |                             ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring bloc
| 7420 |                                 let c = if fault == ElectricalFaultMode::None { Color32::GREEN } else { Color3
| 7421 |                                 ui.label(RichText::new(format!("{}: {}", title, fault)).color(c).strong()); | 
| 7422 |                                 for (label, mode) in [ | iteration logic; may alter timing, throughput, and sa
| 7423 |                                     ("OK", ElectricalFaultMode::None), | support logic; impact depends on neig
| 7424 |                                     ("OPEN", ElectricalFaultMode::OpenCircuit), | support logic; impact depend
| 7425 |                                     ("HI-R", ElectricalFaultMode::HighResistance), | support logic; impact dep
| 7426 |                                 ] { | support logic; impact depends on neighboring block and data flow. |
| 7427 |                                     if ui.small_button(label).clicked() { | runtime gate; condition changes ca
| 7428 |                                         self.cmds.push(Cmd::ElectricalSetControlFault(point, mode)); | support
| 7429 |                                     } | scope delimiter; structural only, but wrong placement changes ownersh
| 7430 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 7431 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 7432 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 7433 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7434 | (blank) | blank separator; no runtime behavior. |
| 7435 |                 ui.columns(2, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 7436 |                     egui::Frame::group(cols[0].style()) | support logic; impact depends on neighboring block a
| 7437 |                         .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and
| 7438 |                         .show(&mut cols[0], \|ui\| { | support logic; impact depends on neighboring block and 
| 7439 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7440 |                                 RichText::new("POWER OVERVIEW") | support logic; impact depends on neighboring
| 7441 |                                     .size(11.0) | support logic; impact depends on neighboring block and data 
| 7442 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends o
| 7443 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7444 |                             let batt_col = if battery_low { | support logic; impact depends on neighboring blo
| 7445 |                                 Color32::RED | support logic; impact depends on neighboring block and data flo
| 7446 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 7447 |                                 Color32::GREEN | support logic; impact depends on neighboring block and data f
| 7448 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 7449 |                             bar_gauge(ui, "Battery", bcm.battery_voltage - 9.0, 7.0, "V", 135.0, batt_col); | 
| 7450 |                             digital_readout(ui, "Batt current", &format!("{:+.1} A", bcm.battery_current_a), C
| 7451 |                             digital_readout(ui, "Alternator", &format!("{:.0} A", bcm.alternator_amps), Color3
| 7452 |                             digital_readout(ui, "ECM branch", if ecm_on { "energized" } else { "off" }, if ecm
| 7453 |                             digital_readout(ui, "TCM branch", if tcm_on { "energized" } else { "off" }, if tcm
| 7454 |                             digital_readout(ui, "Electrical", &format!("{:.2} kW", vcm.power.electrical_demand
| 7455 |                             digital_readout(ui, "Total load", &format!("{:.1} kW", vcm.power.total_load_kw), i
| 7456 |                             digital_readout(ui, "Load shed", if vcm.power.load_shed_active { vcm.power.load_sh
| 7457 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 7458 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7459 |                                 RichText::new("FUSE MATRIX") | support logic; impact depends on neighboring bl
| 7460 |                                     .size(11.0) | support logic; impact depends on neighboring block and data 
| 7461 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends o
| 7462 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7463 |                             ScrollArea::vertical() | support logic; impact depends on neighboring block and da
| 7464 |                                 .id_salt("electrical::fuse_scroll") | support logic; impact depends on neighbo
| 7465 |                                 .max_height(260.0) | support logic; impact depends on neighboring block and da
| 7466 |                                 .auto_shrink([false, false]) | support logic; impact depends on neighboring bl
| 7467 |                                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and da
| 7468 |                                     for fuse in &bcm.fuses { | iteration logic; may alter timing, throughput, 
| 7469 |                                         let gauge = Self::electrical_wire_mm2(fuse.rating_a); | support logic;
| 7470 |                                         let cap_total = 3.0 * Self::electrical_cap_nf_per_m(gauge, false); | s
| 7471 |                                         let ratio = if fuse.rating_a > 0.0 { | support logic; impact depends o
| 7472 |                                             (fuse.load_a / fuse.rating_a).clamp(0.0, 1.4) | support logic; imp
| 7473 |                                         } else { | support logic; impact depends on neighboring block and data
| 7474 |                                             0.0 | support logic; impact depends on neighboring block and data 
| 7475 |                                         }; | scope delimiter; structural only, but wrong placement changes own
| 7476 |                                         let row_col = if fuse.blown { | support logic; impact depends on neigh
| 7477 |                                             Color32::RED | support logic; impact depends on neighboring block 
| 7478 |                                         } else if ratio > 0.9 { | support logic; impact depends on neighboring
| 7479 |                                             Color32::YELLOW | support logic; impact depends on neighboring blo
| 7480 |                                         } else { | support logic; impact depends on neighboring block and data
| 7481 |                                             Color32::GREEN | support logic; impact depends on neighboring bloc
| 7482 |                                         }; | scope delimiter; structural only, but wrong placement changes own
| 7483 |                                         egui::Frame::group(ui.style()) | support logic; impact depends on neig
| 7484 |                                             .fill(Color32::from_gray(20)) | support logic; impact depends on n
| 7485 |                                             .show(ui, \|ui\| { | support logic; impact depends on neighboring 
```

```
| 7486 |                                                ui.horizontal(\|ui\| { | support logic; impact depends on neigh
| 7487 |                                                    ui.label(RichText::new(fuse.id).size(10.6).color(row_col).s
| 7488 |                                                    ui.separator(); | support logic; impact depends on neighbor
| 7489 |                                                    ui.label(RichText::new(format!("{:.1}/{:.1} A", fuse.load_a
| 7490 |                                                    ui.separator(); | support logic; impact depends on neighbor
| 7491 |                                                    ui.label(RichText::new(format!("{:.2} mm2", gauge)).size(9
| 7492 |                                                    ui.separator(); | support logic; impact depends on neighbor
| 7493 |                                                    ui.label(RichText::new(format!("C~{:.2} nF", cap_total)).si
| 7494 |                                                    if fuse.blown { | runtime gate; condition changes can enabl
| 7495 |                                                        ui.separator(); | support logic; impact depends on neig
| 7496 |                                                        ui.label(RichText::new("OPEN").size(9.8).color(Color32
| 7497 |                                                    } | scope delimiter; structural only, but wrong placement c
| 7498 |                                                }); | support logic; impact depends on neighboring block and da
| 7499 |                                                ui.add(ProgressBar::new((ratio / 1.15).clamp(0.0, 1.0) as f32)
| 7500 |                                            }); | support logic; impact depends on neighboring block and data f
| 7501 |                                        } | scope delimiter; structural only, but wrong placement changes ownership
| 7502 |                                    }); | support logic; impact depends on neighboring block and data flow. |
| 7503 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 7504 | (blank) | blank separator; no runtime behavior. |
| 7505 |                        egui::Frame::group(cols[1].style()) | support logic; impact depends on neighboring block an
| 7506 |                            .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and
| 7507 |                            .show(&mut cols[1], \|ui\| { | support logic; impact depends on neighboring block and c
| 7508 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7509 |                                    RichText::new("LIVE ELECTRICAL TOPOLOGY") | support logic; impact depends on ne
| 7510 |                                        .size(11.0) | support logic; impact depends on neighboring block and data f
| 7511 |                                        .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 7512 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 7513 |                                let (rect, _) = ui.allocate_exact_size( | support logic; impact depends on neighbor
| 7514 |                                    Vec2::new(ui.available_width(), 340.0), | support logic; impact depends on neig
| 7515 |                                    Sense::hover(), | support logic; impact depends on neighboring block and data f
| 7516 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 7517 |                                let p = ui.painter_at(rect); | support logic; impact depends on neighboring block a
| 7518 |                                p.rect_filled(rect, 6.0, Color32::from_gray(10)); | support logic; impact depends o
| 7519 |                                p.rect_stroke(rect, 6.0, Stroke::new(1.0, Color32::from_gray(55))); | support logic
| 7520 | (blank) | blank separator; no runtime behavior. |
| 7521 |                                let map = \|x: f32, y: f32\| -> Pos2 { | support logic; impact depends on neighbori
| 7522 |                                    Pos2::new(rect.left() + x * rect.width(), rect.top() + y * rect.height()) | sup
| 7523 |                                }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 7524 |                                let battery = map(0.12, 0.20); | support logic; impact depends on neighboring block
| 7525 |                                let alternator = map(0.12, 0.62); | support logic; impact depends on neighboring bl
| 7526 |                                let fuse_box = map(0.35, 0.20); | support logic; impact depends on neighboring bloc
| 7527 |                                let bcm_p = map(0.56, 0.10); | support logic; impact depends on neighboring block a
| 7528 |                                let vcm_p = map(0.56, 0.33); | support logic; impact depends on neighboring block a
| 7529 |                                let hcm_p = map(0.56, 0.58); | support logic; impact depends on neighboring block a
| 7530 |                                let icm_p = map(0.56, 0.82); | support logic; impact depends on neighboring block a
| 7531 |                                let ecm_p = map(0.83, 0.10); | support logic; impact depends on neighboring block a
| 7532 |                                let tcm_p = map(0.83, 0.33); | support logic; impact depends on neighboring block a
| 7533 |                                let abs_p = map(0.83, 0.58); | support logic; impact depends on neighboring block a
| 7534 |                                let body_p = map(0.35, 0.82); | support logic; impact depends on neighboring block
| 7535 | (blank) | blank separator; no runtime behavior. |
| 7536 |                                let draw_link = \|p: &Painter, a: Pos2, b: Pos2, energized: bool, can_link: bool\|
| 7537 |                                    let col = if can_link { | support logic; impact depends on neighboring block an
| 7538 |                                        if energized { Color32::from_rgb(90, 220, 220) } else { Color32::from_gray(
| 7539 |                                    } else if energized { | support logic; impact depends on neighboring block and
| 7540 |                                        Color32::from_rgb(255, 210, 90) | support logic; impact depends on neighbor
| 7541 |                                    } else { | support logic; impact depends on neighboring block and data flow. |
| 7542 |                                        Color32::from_gray(65) | support logic; impact depends on neighboring block
| 7543 |                                    }; | scope delimiter; structural only, but wrong placement changes ownership/li
| 7544 |                                    p.line_segment([a, b], Stroke::new(if can_link { 2.0 } else { 2.8 }, col)); | s
| 7545 |                                }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 7546 | (blank) | blank separator; no runtime behavior. |
| 7547 |                                let draw_node = \|p: &Painter, pos: Pos2, title: &str, detail: &str, active: bool,
| 7548 |                                    let r = Rect::from_center_size(pos, Vec2::new(96.0, 34.0)); | support logic; im
| 7549 |                                    p.rect_filled(r, 5.0, if active { fill } else { Color32::from_gray(24) }); | su
| 7550 |                                    p.rect_stroke(r, 5.0, Stroke::new(1.0, if active { Color32::WHITE } else { Colo
| 7551 |                                    p.text(r.center_top() + Vec2::new(0.0, 4.0), Align2::CENTER_TOP, title, FontId
| 7552 |                                    p.text(r.center_bottom() - Vec2::new(0.0, 4.0), Align2::CENTER_BOTTOM, detail,
| 7553 |                                }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 7554 | (blank) | blank separator; no runtime behavior. |
| 7555 |                                draw_link(&p, battery, fuse_box, branches.iter().any(\|b\| b.id == "BAT-MAIN" && b
| 7556 |                                draw_link(&p, alternator, battery, branches.iter().any(\|b\| b.id == "ALT-B+" && b
| 7557 |                                draw_link(&p, fuse_box, bcm_p, branches.iter().any(\|b\| b.id == "BCM-PWR" && b.ene
| 7558 |                                draw_link(&p, fuse_box, vcm_p, branches.iter().any(\|b\| b.id == "VCM-PWR" && b.ene
| 7559 |                                draw_link(&p, fuse_box, hcm_p, branches.iter().any(\|b\| b.id == "HCM-PWR" && b.ene
| 7560 |                                draw_link(&p, fuse_box, body_p, branches.iter().any(\|b\| b.id == "BODY-LOAD" && b
| 7561 |                                draw_link(&p, bcm_p, icm_p, branches.iter().any(\|b\| b.id == "ICM-PWR" && b.energi
```

```
| 7562 |                            draw_link(&p, vcm_p, ecm_p, branches.iter().any(\|b\| b.id == "ECM-PWR" && b.energi
| 7563 |                            draw_link(&p, vcm_p, tcm_p, branches.iter().any(\|b\| b.id == "TCM-PWR" && b.energi
| 7564 |                            draw_link(&p, vcm_p, abs_p, branches.iter().any(\|b\| b.id == "ABS-PWR" && b.energi
| 7565 |                            draw_link(&p, bcm_p, vcm_p, branches.iter().any(\|b\| b.id == "MS-CAN-BODY" && b.en
| 7566 |                            draw_link(&p, vcm_p, ecm_p, branches.iter().any(\|b\| b.id == "HS-CAN-1" && b.energ
| 7567 |                            draw_link(&p, vcm_p, tcm_p, branches.iter().any(\|b\| b.id == "HS-CAN-1" && b.energ
| 7568 |                            draw_link(&p, vcm_p, abs_p, branches.iter().any(\|b\| b.id == "HS-CAN-1" && b.energ
| 7569 |                            draw_link(&p, vcm_p, hcm_p, branches.iter().any(\|b\| b.id == "HS-CAN-1" && b.energ
| 7570 |                            draw_link(&p, bcm_p, icm_p, branches.iter().any(\|b\| b.id == "MS-CAN-BODY" && b.en
| 7571 | (blank) | blank separator; no runtime behavior. |
| 7572 |                            draw_node(&p, battery, "BATTERY", &format!("{:.1} V / {:.0}%", bcm.battery_voltage,
| 7573 |                            draw_node(&p, alternator, "ALTERNATOR", &format!("{:.0} A", bcm.alternator_amps), b
| 7574 |                            draw_node(&p, fuse_box, "MAIN FUSE BOX", if any_blown { "open / short detected" } e
| 7575 |                            draw_node(&p, bcm_p, "BCM", &format!("{:.0} A load", bcm.total_load_amps), bcm_on,
| 7576 |                            draw_node(&p, vcm_p, "VCM", if vcm.power.load_shed_active { "load shed" } else { "p
| 7577 |                            draw_node(&p, hcm_p, "HCM", if hcm_on { "hyd online" } else { "offline" }, hcm_on,
| 7578 |                            draw_node(&p, icm_p, "ICM", if icm_on { "cluster online" } else { "offline" }, icm_
| 7579 |                            draw_node(&p, ecm_p, "ECM", if ecm_on { "engine online" } else { "offline" }, ecm_o
| 7580 |                            draw_node(&p, tcm_p, "TCM", if tcm_on { "trans online" } else { "offline" }, tcm_on
| 7581 |                            draw_node(&p, abs_p, "ABS/ESP", if abs_on { "brake online" } else { "offline" }, ab
| 7582 |                            draw_node(&p, body_p, "BODY LOADS", &format!("{:.0} A", body_load_a), body_load_a >
| 7583 | (blank) | blank separator; no runtime behavior. |
| 7584 |                            p.text(map(0.52, 0.03), Align2::CENTER_TOP, "yellow = power branch \| cyan = CAN ba
| 7585 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 7586 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 7587 | (blank) | blank separator; no runtime behavior. |
| 7588 |                    ui.add_space(8.0); | support logic; impact depends on neighboring block and data flow. |
| 7589 |                    egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data fl
| 7590 |                        .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data
| 7591 |                        .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 7592 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7593 |                                RichText::new("HARNESS / ELECTRICAL BRANCH REFERENCE") | support logic; impact depe
| 7594 |                                    .size(11.0) | support logic; impact depends on neighboring block and data flow
| 7595 |                                    .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on nei
| 7596 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 7597 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7598 |                                RichText::new("Gauge, resistance, capacitance and drop below are computed from the
| 7599 |                                    .size(9.6) | support logic; impact depends on neighboring block and data flow.
| 7600 |                                    .color(Color32::from_gray(160)), | support logic; impact depends on neighboring
| 7601 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 7602 |                            ScrollArea::vertical() | support logic; impact depends on neighboring block and data fl
| 7603 |                                .id_salt("electrical::branch_scroll") | support logic; impact depends on neighborin
| 7604 |                                .max_height(290.0) | support logic; impact depends on neighboring block and data fl
| 7605 |                                .auto_shrink([false, false]) | support logic; impact depends on neighboring block a
| 7606 |                                .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data fl
| 7607 |                                    for branch in &branches { | iteration logic; may alter timing, throughput, and
| 7608 |                                        let cap_total = branch.capacitance_nf; | support logic; impact depends on n
| 7609 |                                        let branch_col = if branch.energized { | support logic; impact depends on n
| 7610 |                                            if branch.line == auto_breaking::ElectricalLine::HsCan \|\| branch.line
| 7611 |                                                Color32::from_rgb(90, 220, 220) | support logic; impact depends on
| 7612 |                                            } else { | support logic; impact depends on neighboring block and data
| 7613 |                                                Color32::from_rgb(255, 210, 90) | support logic; impact depends on
| 7614 |                                            } | scope delimiter; structural only, but wrong placement changes owner
| 7615 |                                        } else { | support logic; impact depends on neighboring block and data flow
| 7616 |                                            Color32::from_gray(90) | support logic; impact depends on neighboring b
| 7617 |                                        }; | scope delimiter; structural only, but wrong placement changes ownership
| 7618 |                                        ui.push_id(branch.id, \|ui\| { | support logic; impact depends on neighbori
| 7619 |                                            egui::Frame::group(ui.style()) | support logic; impact depends on neigh
| 7620 |                                                .fill(Color32::from_gray(20)) | support logic; impact depends on ne
| 7621 |                                                .show(ui, \|ui\| { | support logic; impact depends on neighboring b
| 7622 |                                                    ui.horizontal_wrapped(\|ui\| { | support logic; impact depends
| 7623 |                                                        ui.label(RichText::new(branch.id).size(10.5).color(branch_c
| 7624 |                                                        ui.separator(); | support logic; impact depends on neighbor
| 7625 |                                                        ui.label(RichText::new(format!("{} -> {}", branch.from, bra
| 7626 |                                                        ui.separator(); | support logic; impact depends on neighbor
| 7627 |                                                        ui.label(RichText::new(format!("Line {}", branch.line)).siz
| 7628 |                                                        ui.separator(); | support logic; impact depends on neighbor
| 7629 |                                                        ui.label(RichText::new(format!("Fuse {}", branch.fuse_id))
| 7630 |                                                        ui.separator(); | support logic; impact depends on neighbor
| 7631 |                                                        ui.label(RichText::new(format!("{}", branch.fault)).size(9
| 7632 |                                                    }); | support logic; impact depends on neighboring block and da
| 7633 |                                                    ui.label( | support logic; impact depends on neighboring block
| 7634 |                                                        RichText::new(format!( | support logic; impact depends on n
| 7635 |                                                            "Ireq={:.2} A \| Iact={:.2} A \| V={:.2} V \| R={:.4} o
| 7636 |                                                            branch.requested_current_a, | support logic; impact dep
| 7637 |                                                            branch.actual_current_a, | support logic; impact depend
```

```
| 7638 |                                                         branch.voltage_v, | support logic; impact depends on ne
| 7639 |                                                         branch.resistance_ohm, | support logic; impact depends
| 7640 |                                                         branch.gauge_mm2, | support logic; impact depends on ne
| 7641 |                                                         branch.length_m, | support logic; impact depends on nei
| 7642 |                                                         cap_total, | support logic; impact depends on neighbori
| 7643 |                                                         branch.note | support logic; impact depends on neighbor
| 7644 |                                                     )) | support logic; impact depends on neighboring block and
| 7645 |                                                     .size(9.6) | support logic; impact depends on neighboring b
| 7646 |                                                     .monospace() | support logic; impact depends on neighboring
| 7647 |                                                     .color(Color32::from_gray(180)), | support logic; impact de
| 7648 |                                                 ); | support logic; impact depends on neighboring block and dat
| 7649 |                                                 ui.horizontal_wrapped(\|ui\| { | support logic; impact depends
| 7650 |                                                     for (label, mode) in [ | iteration logic; may alter timing,
| 7651 |                                                         ("OK", ElectricalFaultMode::None), | support logic; imp
| 7652 |                                                         ("OPEN", ElectricalFaultMode::OpenCircuit), | support
| 7653 |                                                         ("SHORT", ElectricalFaultMode::ShortToGround), | suppo
| 7654 |                                                         ("HI-R", ElectricalFaultMode::HighResistance), | suppo
| 7655 |                                                     ] { | support logic; impact depends on neighboring block a
| 7656 |                                                         if ui.small_button(label).clicked() { | runtime gate; c
| 7657 |                                                             self.cmds.push(Cmd::ElectricalSetBranchFault(branch
| 7658 |                                                         } | scope delimiter; structural only, but wrong placeme
| 7659 |                                                     } | scope delimiter; structural only, but wrong placement c
| 7660 |                                                     if branch.blown { | runtime gate; condition changes can ena
| 7661 |                                                         ui.label(RichText::new("branch fuse blown").size(9.6).c
| 7662 |                                                     } | scope delimiter; structural only, but wrong placement c
| 7663 |                                                 }); | support logic; impact depends on neighboring block and da
| 7664 |                                             }); | support logic; impact depends on neighboring block and data f
| 7665 |                                         }); | support logic; impact depends on neighboring block and data flow. |
| 7666 |                                     } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 7667 |                                 }); | support logic; impact depends on neighboring block and data flow. |
| 7668 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 7669 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 7670 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7671 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7672 | (blank) | blank separator; no runtime behavior. |
| 7673 | // ━━━
```

```
| 7714 |                     )) | support logic; impact depends on neighboring block and data flow. |
| 7715 |                     .size(10.8) | support logic; impact depends on neighboring block and data flow. |
| 7716 |                     .color(if ok { Color32::GREEN } else { Color32::YELLOW }), | support logic; impact depends
| 7717 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 7718 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7719 |         }); | support logic; impact depends on neighboring block and data flow. |
| 7720 |         ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 7721 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 7722 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 7723 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 7724 |                 ui.columns(2, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 7725 |                     // GPS | comment/documentation; changing affects understanding and intent traceability. |
| 7726 |                     egui::Frame::group(cols[0].style()) | support logic; impact depends on neighboring block a
| 7727 |                         .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and
| 7728 |                         .show(&mut cols[0], \|ui\| { | support logic; impact depends on neighboring block and
| 7729 |                             let g = &self.bench.gps; | support logic; impact depends on neighboring block and
| 7730 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7731 |                                 RichText::new("■ GPS/GNSS — u-blox ZED-F9P") | support logic; impact depends o
| 7732 |                                     .size(12.0) | support logic; impact depends on neighboring block and data
| 7733 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends o
| 7734 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7735 |                             let fc = match g.fix_quality { | support logic; impact depends on neighboring bloc
| 7736 |                                 auto_breaking::gps::GpsFixQuality::RtkFix => Color32::GREEN, | support logic;
| 7737 |                                 auto_breaking::gps::GpsFixQuality::DgpsFix => { | support logic; impact depend
| 7738 |                                     Color32::from_rgb(100, 220, 100) | support logic; impact depends on neighb
| 7739 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/li
| 7740 |                                 auto_breaking::gps::GpsFixQuality::SpsFix => Color32::YELLOW, | support logic;
| 7741 |                                 _ => Color32::RED, | support logic; impact depends on neighboring block and da
| 7742 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 7743 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7744 |                                 RichText::new(format!("Fix: {}", g.fix_quality)) | support logic; impact depen
| 7745 |                                     .size(12.0) | support logic; impact depends on neighboring block and data
| 7746 |                                     .color(fc) | support logic; impact depends on neighboring block and data f
| 7747 |                                     .strong(), | support logic; impact depends on neighboring block and data f
| 7748 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7749 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 7750 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7751 |                                 "Position", | support logic; impact depends on neighboring block and data flow
| 7752 |                                 &g.position_string(), | support logic; impact depends on neighboring block and
| 7753 |                                 Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and
| 7754 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7755 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 7756 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7757 |                                 "Altitude", | support logic; impact depends on neighboring block and data flow
| 7758 |                                 &format!("{:.1} m MSL", g.altitude_msl), | support logic; impact depends on ne
| 7759 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block
| 7760 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7761 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 7762 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7763 |                                 "Speed", | support logic; impact depends on neighboring block and data flow. |
| 7764 |                                 &format!("{:.2} km/h  {:.1}°", g.speed_kmh, g.course_deg), | support logic; im
| 7765 |                                 Color32::GREEN, | support logic; impact depends on neighboring block and data
| 7766 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7767 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 7768 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7769 |                                 "HDOP", | support logic; impact depends on neighboring block and data flow. |
| 7770 |                                 &format!("{:.2}  PDOP:{:.2}", g.hdop, g.pdop), | support logic; impact depends
| 7771 |                                 Color32::from_gray(180), | support logic; impact depends on neighboring block
| 7772 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7773 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 7774 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7775 |                                 "Sats", | support logic; impact depends on neighboring block and data flow. |
| 7776 |                                 &format!("{} used / {} view", g.sats_used, g.sats_in_view), | support logic; i
| 7777 |                                 Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 7778 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7779 |                             digital_readout( | support logic; impact depends on neighboring block and data flo
| 7780 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7781 |                                 "H-Acc", | support logic; impact depends on neighboring block and data flow. |
| 7782 |                                 &format!("{:.2} m (1σ)", g.hacc_m), | support logic; impact depends on neighbo
| 7783 |                                 Color32::from_rgb(100, 200, 100), | support logic; impact depends on neighborin
| 7784 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7785 |                             digital_readout(ui, "UTC", &g.utc_time, Color32::from_gray(155)); | support logic;
| 7786 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 7787 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7788 |                                 RichText::new("LATEST NMEA:") | support logic; impact depends on neighboring b
| 7789 |                                     .size(10.5) | support logic; impact depends on neighboring block and data
```

```
| 7790 |                                     .color(Color32::from_gray(130)), | support logic; impact depends on neighbo
| 7791 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 7792 |                                 for nmea in g.nmea_queue.iter().take(4) { | iteration logic; may alter timing, thro
| 7793 |                                     ui.label( | support logic; impact depends on neighboring block and data flow.
| 7794 |                                         RichText::new(nmea.trim()) | support logic; impact depends on neighboring b
| 7795 |                                             .size(9.5) | support logic; impact depends on neighboring block and dat
| 7796 |                                             .monospace() | support logic; impact depends on neighboring block and d
| 7797 |                                             .color(Color32::from_gray(180)), | support logic; impact depends on nei
| 7798 |                                     ); | support logic; impact depends on neighboring block and data flow. |
| 7799 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 7800 |                                 ui.separator(); | support logic; impact depends on neighboring block and data flow
| 7801 |                                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7802 |                                     RichText::new("SATELLITES:") | support logic; impact depends on neighboring blo
| 7803 |                                         .size(10.5) | support logic; impact depends on neighboring block and data f
| 7804 |                                         .color(Color32::from_gray(130)), | support logic; impact depends on neighbo
| 7805 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 7806 |                                 for sat in &g.satellites { | iteration logic; may alter timing, throughput, and sat
| 7807 |                                     let c = if !sat.used { | support logic; impact depends on neighboring block and
| 7808 |                                         Color32::from_gray(55) | support logic; impact depends on neighboring block
| 7809 |                                     } else if sat.snr > 35.0 { | support logic; impact depends on neighboring block
| 7810 |                                         Color32::GREEN | support logic; impact depends on neighboring block and dat
| 7811 |                                     } else { | support logic; impact depends on neighboring block and data flow. |
| 7812 |                                         Color32::YELLOW | support logic; impact depends on neighboring block and da
| 7813 |                                     }; | scope delimiter; structural only, but wrong placement changes ownership/li
| 7814 |                                     let bar: String = | support logic; impact depends on neighboring block and data
| 7815 |                                         "■".repeat(((sat.snr / 52.0 * 12.0) as usize).min(12)); | support logic; i
| 7816 |                                     ui.label( | support logic; impact depends on neighboring block and data flow.
| 7817 |                                         RichText::new(format!( | support logic; impact depends on neighboring block
| 7818 |                                             "PRN{:02} {} El:{:2.0}° Az:{:3.0}° SNR:{:4.1} {} {}", | support logic;
| 7819 |                                             sat.prn, | support logic; impact depends on neighboring block and data
| 7820 |                                             sat.constellation, | support logic; impact depends on neighboring block
| 7821 |                                             sat.elevation, | support logic; impact depends on neighboring block and
| 7822 |                                             sat.azimuth, | support logic; impact depends on neighboring block and d
| 7823 |                                             sat.snr, | support logic; impact depends on neighboring block and data
| 7824 |                                             bar, | support logic; impact depends on neighboring block and data flo
| 7825 |                                             if sat.used { "✓" } else { "·" } | runtime gate; condition changes can
| 7826 |                                         )) | support logic; impact depends on neighboring block and data flow. |
| 7827 |                                         .size(9.5) | support logic; impact depends on neighboring block and data f
| 7828 |                                         .monospace() | support logic; impact depends on neighboring block and data
| 7829 |                                         .color(c), | support logic; impact depends on neighboring block and data f
| 7830 |                                     ); | support logic; impact depends on neighboring block and data flow. |
| 7831 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 7832 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 7833 |                         // IMU | comment/documentation; changing affects understanding and intent traceability. |
| 7834 |                         egui::Frame::group(cols[1].style()) | support logic; impact depends on neighboring block a
| 7835 |                             .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and
| 7836 |                             .show(&mut cols[1], \|ui\| { | support logic; impact depends on neighboring block and d
| 7837 |                                 let imu = &self.bench.imu; | support logic; impact depends on neighboring block and
| 7838 |                                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7839 |                                     RichText::new("■ IMU — Madgwick AHRS 9-DOF") | support logic; impact depends o
| 7840 |                                         .size(12.0) | support logic; impact depends on neighboring block and data f
| 7841 |                                         .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 7842 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 7843 |                                 ui.columns(3, \|c\| { | support logic; impact depends on neighboring block and data
| 7844 |                                     egui::Frame::dark_canvas(c[0].style()).show(&mut c[0], \|ui\| { | support logic
| 7845 |                                         arc_gauge( | support logic; impact depends on neighboring block and data f
| 7846 |                                             ui, | support logic; impact depends on neighboring block and data flow
| 7847 |                                             imu.roll_deg, | support logic; impact depends on neighboring block and
| 7848 |                                             -180.0, | support logic; impact depends on neighboring block and data f
| 7849 |                                             180.0, | support logic; impact depends on neighboring block and data f
| 7850 |                                             "ROLL", | support logic; impact depends on neighboring block and data f
| 7851 |                                             "°", | support logic; impact depends on neighboring block and data flo
| 7852 |                                             95.0, | support logic; impact depends on neighboring block and data flo
| 7853 |                                             25.0, | support logic; impact depends on neighboring block and data flo
| 7854 |                                             45.0, | support logic; impact depends on neighboring block and data flo
| 7855 |                                             Some(-25.0), | support logic; impact depends on neighboring block and d
| 7856 |                                         ) | support logic; impact depends on neighboring block and data flow. |
| 7857 |                                     }); | support logic; impact depends on neighboring block and data flow. |
| 7858 |                                     egui::Frame::dark_canvas(c[1].style()).show(&mut c[1], \|ui\| { | support logic
| 7859 |                                         arc_gauge( | support logic; impact depends on neighboring block and data f
| 7860 |                                             ui, | support logic; impact depends on neighboring block and data flow
| 7861 |                                             imu.pitch_deg, | support logic; impact depends on neighboring block and
| 7862 |                                             -90.0, | support logic; impact depends on neighboring block and data f
| 7863 |                                             90.0, | support logic; impact depends on neighboring block and data flo
| 7864 |                                             "PITCH", | support logic; impact depends on neighboring block and data
| 7865 |                                             "°", | support logic; impact depends on neighboring block and data flo
```

```
| 7866 |                                         95.0, | support logic; impact depends on neighboring block and data flo
| 7867 |                                         15.0, | support logic; impact depends on neighboring block and data flo
| 7868 |                                         30.0, | support logic; impact depends on neighboring block and data flo
| 7869 |                                         Some(-15.0), | support logic; impact depends on neighboring block and d
| 7870 |                                     ) | support logic; impact depends on neighboring block and data flow. |
| 7871 |                                 }); | support logic; impact depends on neighboring block and data flow. |
| 7872 |                                 egui::Frame::dark_canvas(c[2].style()).show(&mut c[2], \|ui\| { | support logic
| 7873 |                                     arc_gauge( | support logic; impact depends on neighboring block and data fl
| 7874 |                                         ui, | support logic; impact depends on neighboring block and data flow
| 7875 |                                         imu.yaw_deg, | support logic; impact depends on neighboring block and d
| 7876 |                                         0.0, | support logic; impact depends on neighboring block and data flow
| 7877 |                                         360.0, | support logic; impact depends on neighboring block and data fl
| 7878 |                                         "YAW/HDG", | support logic; impact depends on neighboring block and dat
| 7879 |                                         "°", | support logic; impact depends on neighboring block and data flow
| 7880 |                                         95.0, | support logic; impact depends on neighboring block and data flo
| 7881 |                                         0.0, | support logic; impact depends on neighboring block and data flow
| 7882 |                                         0.0, | support logic; impact depends on neighboring block and data flow
| 7883 |                                         None, | support logic; impact depends on neighboring block and data flo
| 7884 |                                     ) | support logic; impact depends on neighboring block and data flow. |
| 7885 |                                 }); | support logic; impact depends on neighboring block and data flow. |
| 7886 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 7887 |                             let w = 110.0; | support logic; impact depends on neighboring block and data flow.
| 7888 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7889 |                                 RichText::new("ACCELEROMETER m/s²:") | support logic; impact depends on neighbo
| 7890 |                                     .size(10.5) | support logic; impact depends on neighboring block and data f
| 7891 |                                     .color(Color32::from_gray(130)), | support logic; impact depends on neighbo
| 7892 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7893 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 7894 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7895 |                                 "Ax (fwd)", | support logic; impact depends on neighboring block and data flow
| 7896 |                                 imu.accel_x, | support logic; impact depends on neighboring block and data flow
| 7897 |                                 20.0, | support logic; impact depends on neighboring block and data flow. |
| 7898 |                                 "m/s²", | support logic; impact depends on neighboring block and data flow. |
| 7899 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 7900 |                                 Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighboring
| 7901 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7902 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 7903 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7904 |                                 "Ay (lat)", | support logic; impact depends on neighboring block and data flow
| 7905 |                                 imu.accel_y.abs(), | support logic; impact depends on neighboring block and dat
| 7906 |                                 20.0, | support logic; impact depends on neighboring block and data flow. |
| 7907 |                                 "m/s²", | support logic; impact depends on neighboring block and data flow. |
| 7908 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 7909 |                                 Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighboring
| 7910 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7911 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 7912 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7913 |                                 "Az (up)", | support logic; impact depends on neighboring block and data flow.
| 7914 |                                 imu.accel_z.abs(), | support logic; impact depends on neighboring block and dat
| 7915 |                                 15.0, | support logic; impact depends on neighboring block and data flow. |
| 7916 |                                 "m/s²", | support logic; impact depends on neighboring block and data flow. |
| 7917 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 7918 |                                 Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighboring
| 7919 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7920 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7921 |                                 RichText::new("G-FORCES:") | support logic; impact depends on neighboring block
| 7922 |                                     .size(10.5) | support logic; impact depends on neighboring block and data f
| 7923 |                                     .color(Color32::from_gray(130)), | support logic; impact depends on neighbo
| 7924 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7925 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 7926 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7927 |                                 "Long G", | support logic; impact depends on neighboring block and data flow.
| 7928 |                                 imu.longitudinal_g.abs(), | support logic; impact depends on neighboring block
| 7929 |                                 1.5, | support logic; impact depends on neighboring block and data flow. |
| 7930 |                                 "g", | support logic; impact depends on neighboring block and data flow. |
| 7931 |                                 w, | support logic; impact depends on neighboring block and data flow. |
| 7932 |                                 Color32::RED, | support logic; impact depends on neighboring block and data flo
| 7933 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7934 |                             bar_gauge(ui, "Lat G", imu.lateral_g.abs(), 1.5, "g", w, Color32::RED); | support l
| 7935 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 7936 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7937 |                                 "IMU Temp", | support logic; impact depends on neighboring block and data flow
| 7938 |                                 &format!("{:.1}°C", imu.temperature_c), | support logic; impact depends on neig
| 7939 |                                 Color32::from_gray(180), | support logic; impact depends on neighboring block a
| 7940 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7941 |                             if imu.accel_fault { | runtime gate; condition changes can enable/disable critical
```

```
| 7942 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 7943 |                                     RichText::new("■ ACCEL FAULT") | support logic; impact depends on neighbor
| 7944 |                                         .size(11.0) | support logic; impact depends on neighboring block and da
| 7945 |                                         .color(Color32::RED), | support logic; impact depends on neighboring bl
| 7946 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 7947 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 7948 |                             if imu.gyro_fault { | runtime gate; condition changes can enable/disable critical p
| 7949 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 7950 |                                     RichText::new("■ GYRO FAULT").size(11.0).color(Color32::RED), | support lo
| 7951 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 7952 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 7953 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 7954 |                             // RADAR summary | comment/documentation; changing affects understanding and intent
| 7955 |                             let r = &self.bench.radar; | support logic; impact depends on neighboring block and
| 7956 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 7957 |                                 RichText::new("■ RADAR 77GHz") | support logic; impact depends on neighboring
| 7958 |                                     .size(12.0) | support logic; impact depends on neighboring block and data
| 7959 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 7960 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7961 |                             let ttc_col = if r.ttc_front < 2.0 { | support logic; impact depends on neighboring
| 7962 |                                 Color32::RED | support logic; impact depends on neighboring block and data flow
| 7963 |                             } else if r.ttc_front < 4.0 { | support logic; impact depends on neighboring block
| 7964 |                                 Color32::YELLOW | support logic; impact depends on neighboring block and data
| 7965 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 7966 |                                 Color32::GREEN | support logic; impact depends on neighboring block and data f
| 7967 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 7968 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 7969 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7970 |                                 "Front TTC", | support logic; impact depends on neighboring block and data flow
| 7971 |                                 &format!("{:.1}s", r.ttc_front.min(99.9)), | support logic; impact depends on
| 7972 |                                 ttc_col, | support logic; impact depends on neighboring block and data flow. |
| 7973 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7974 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 7975 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7976 |                                 "Lead dist", | support logic; impact depends on neighboring block and data flow
| 7977 |                                 &format!("{:.1}m", r.closest_front_m.min(999.0)), | support logic; impact depe
| 7978 |                                 Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and
| 7979 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7980 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 7981 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7982 |                                 "BSM L/R", | support logic; impact depends on neighboring block and data flow.
| 7983 |                                 &format!( | support logic; impact depends on neighboring block and data flow.
| 7984 |                                     "{}/{}", | support logic; impact depends on neighboring block and data flow
| 7985 |                                     if r.bsm_left { "■" } else { "✓" }, | runtime gate; condition changes can
| 7986 |                                     if r.bsm_right { "■" } else { "✓" } | runtime gate; condition changes can
| 7987 |                                 ), | support logic; impact depends on neighboring block and data flow. |
| 7988 |                                 if r.bsm_left \|\| r.bsm_right { | runtime gate; condition changes can enable/
| 7989 |                                     Color32::YELLOW | support logic; impact depends on neighboring block and da
| 7990 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 7991 |                                     Color32::GREEN | support logic; impact depends on neighboring block and da
| 7992 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 7993 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 7994 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 7995 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 7996 |                                 "Targets", | support logic; impact depends on neighboring block and data flow.
| 7997 |                                 &format!("{}", r.total_targets()), | support logic; impact depends on neighbor
| 7998 |                                 Color32::from_gray(180), | support logic; impact depends on neighboring block a
| 7999 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8000 |                             for t in r.front_center.targets.iter().take(5) { | iteration logic; may alter timi
| 8001 |                                 let c = if t.ttc_s < 2.0 { | support logic; impact depends on neighboring bloc
| 8002 |                                     Color32::RED | support logic; impact depends on neighboring block and data
| 8003 |                                 } else if t.ttc_s < 4.0 { | support logic; impact depends on neighboring block
| 8004 |                                     Color32::YELLOW | support logic; impact depends on neighboring block and da
| 8005 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 8006 |                                     Color32::GREEN | support logic; impact depends on neighboring block and da
| 8007 |                                 }; | scope delimiter; structural only, but wrong placement changes ownership/li
| 8008 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 8009 |                                     RichText::new(format!( | support logic; impact depends on neighboring bloc
| 8010 |                                         "ID{:02} {:5.1}m Az:{:+.1}° {:.1}m/s TTC:{:.1}s {}", | support logic;
| 8011 |                                         t.id, | support logic; impact depends on neighboring block and data flo
| 8012 |                                         t.range_m, | support logic; impact depends on neighboring block and da
| 8013 |                                         t.azimuth_deg, | support logic; impact depends on neighboring block an
| 8014 |                                         -t.range_rate_ms, | support logic; impact depends on neighboring block
| 8015 |                                         t.ttc_s.min(99.9), | support logic; impact depends on neighboring bloc
| 8016 |                                         t.object_class | support logic; impact depends on neighboring block an
| 8017 |                                     )) | support logic; impact depends on neighboring block and data flow. |
```

```
| 8018 |                                     .size(10.0) | support logic; impact depends on neighboring block and data f
| 8019 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 8020 |                                     .color(c), | support logic; impact depends on neighboring block and data fl
| 8021 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 8022 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8023 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8024 |                             let l = &self.bench.lidar; | support logic; impact depends on neighboring block and
| 8025 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8026 |                                 RichText::new("■ LIDAR VLP-16") | support logic; impact depends on neighboring
| 8027 |                                     .size(12.0) | support logic; impact depends on neighboring block and data f
| 8028 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 8029 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8030 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8031 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8032 |                                 "Pts/scan", | support logic; impact depends on neighboring block and data flow
| 8033 |                                 &format!("{}", l.points_per_scan), | support logic; impact depends on neighbori
| 8034 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block a
| 8035 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8036 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8037 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8038 |                                 "Clusters", | support logic; impact depends on neighboring block and data flow
| 8039 |                                 &format!("{}", l.obstacles_seen), | support logic; impact depends on neighbori
| 8040 |                                 Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 8041 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8042 |                             for cl in l.clusters.iter().take(4) { | iteration logic; may alter timing, through
| 8043 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 8044 |                                     RichText::new(format!( | support logic; impact depends on neighboring bloc
| 8045 |                                         "ID{:03} {} {:.1}m  {:.1}×{:.1}m  {}pts", | support logic; impact depe
| 8046 |                                         cl.id, | support logic; impact depends on neighboring block and data f
| 8047 |                                         cl.object_type, | support logic; impact depends on neighboring block a
| 8048 |                                         cl.distance_m, | support logic; impact depends on neighboring block an
| 8049 |                                         cl.length_m, | support logic; impact depends on neighboring block and
| 8050 |                                         cl.width_m, | support logic; impact depends on neighboring block and da
| 8051 |                                         cl.point_count | support logic; impact depends on neighboring block an
| 8052 |                                     )) | support logic; impact depends on neighboring block and data flow. |
| 8053 |                                     .size(10.0) | support logic; impact depends on neighboring block and data f
| 8054 |                                     .monospace() | support logic; impact depends on neighboring block and data
| 8055 |                                     .color(Color32::from_gray(190)), | support logic; impact depends on neighb
| 8056 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 8057 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8058 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 8059 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 8060 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 8061 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8062 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8063 | (blank) | blank separator; no runtime behavior. |
| 8064 | // ■■■
```

```
| 8094 |         let lt_s = format!("{}", self.bench.ad.lane.left_type); | support logic; impact depends on neighboring
| 8095 |         let rt_s = format!("{}", self.bench.ad.lane.right_type); | support logic; impact depends on neighborin
| 8096 |         let safe_s = format!("{}", self.bench.ad.safety_status); | support logic; impact depends on neighborin
| 8097 |         let deg_r = self.bench.ad.degrade_reason; | support logic; impact depends on neighboring block and data
| 8098 |         let drows = self.bench.ad.drowsiness_pct; | support logic; impact depends on neighboring block and data
| 8099 |         let hands = self.bench.ad.hands_off_s; | support logic; impact depends on neighboring block and data f
| 8100 |         let alert = self.bench.ad.driver_alert; | support logic; impact depends on neighboring block and data
| 8101 |         let _ego_spd = self.bench.tcm.ground_speed_kmh; | support logic; impact depends on neighboring block a
| 8102 |         let path_v: Vec<_> = self | support logic; impact depends on neighboring block and data flow. |
| 8103 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 8104 |             .ad | support logic; impact depends on neighboring block and data flow. |
| 8105 |             .planned_path | support logic; impact depends on neighboring block and data flow. |
| 8106 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 8107 |             .map(\|w\| (w.x, w.y)) | support logic; impact depends on neighboring block and data flow. |
| 8108 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 8109 | (blank) | blank separator; no runtime behavior. |
| 8110 |         let mut do_engage = false; | support logic; impact depends on neighboring block and data flow. |
| 8111 |         let mut do_lka = false; | support logic; impact depends on neighboring block and data flow. |
| 8112 |         let mut do_acc_up = false; | support logic; impact depends on neighboring block and data flow. |
| 8113 |         let mut do_acc_dn = false; | support logic; impact depends on neighboring block and data flow. |
| 8114 |         let mut do_hdw_up = false; | support logic; impact depends on neighboring block and data flow. |
| 8115 |         let mut do_hdw_dn = false; | support logic; impact depends on neighboring block and data flow. |
| 8116 |         let readiness = self.ad_readiness_issues(); | support logic; impact depends on neighboring block and d
| 8117 |         let can_engage_ad = self.bench.engine_running() && !matches!(self.bench.tcm.direction, Direction::Park
| 8118 | (blank) | blank separator; no runtime behavior. |
| 8119 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 8120 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 8121 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 8122 |                 ui.columns(3, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 8123 |                     // ■■ Left: Controls ■■■
```

```
| 8170 |                                Button::new( | support logic; impact depends on neighboring block and data flo
| 8171 |                                    RichText::new(if eng { | support logic; impact depends on neighboring bloc
| 8172 |                                        "■ DISENGAGE AD" | support logic; impact depends on neighboring block
| 8173 |                                    } else { | support logic; impact depends on neighboring block and data flo
| 8174 |                                        "■ ENGAGE AD" | support logic; impact depends on neighboring block and
| 8175 |                                    }) | support logic; impact depends on neighboring block and data flow. |
| 8176 |                                    .size(13.0), | support logic; impact depends on neighboring block and data
| 8177 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 8178 |                                .fill(fill) | support logic; impact depends on neighboring block and data flow
| 8179 |                                .min_size(Vec2::new(180.0, 34.0)), | support logic; impact depends on neighbor
| 8180 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8181 |                            if !eng && !can_engage_ad { | runtime gate; condition changes can enable/disable c
| 8182 |                                engage_resp = engage_resp.on_disabled_hover_text( | support logic; impact depe
| 8183 |                                    "Para engatar o AD, deixe o motor em RUN e saia de Park.", | support logic
| 8184 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8185 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 8186 |                            if engage_resp.clicked() { | runtime gate; condition changes can enable/disable cr
| 8187 |                                do_engage = true; | support logic; impact depends on neighboring block and data
| 8188 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 8189 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8190 |                                RichText::new(if lead.is_finite() { | support logic; impact depends on neighbo
| 8191 |                                    "ACC esta usando alvo frontal e TTC/THW para modular torque/freio." | suppo
| 8192 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 8193 |                                    "Sem alvo frontal: ACC acelera ate a velocidade configurada e LKA depende
| 8194 |                                }) | support logic; impact depends on neighboring block and data flow. |
| 8195 |                                .size(10.0) | support logic; impact depends on neighboring block and data flow
| 8196 |                                .color(Color32::from_gray(175)), | support logic; impact depends on neighboring
| 8197 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8198 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8199 |                            // ACC | comment/documentation; changing affects understanding and intent traceabi
| 8200 |                            let ac = match acc_s { | support logic; impact depends on neighboring block and da
| 8201 |                                FeatureState::Active => Color32::GREEN, | support logic; impact depends on nei
| 8202 |                                FeatureState::Ready => Color32::YELLOW, | support logic; impact depends on nei
| 8203 |                                _ => Color32::from_gray(80), | support logic; impact depends on neighboring bl
| 8204 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 8205 |                            ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and da
| 8206 |                                ui.label(RichText::new("ACC:").size(12.0).color(ac).strong()); | support logic
| 8207 |                                ui.label( | support logic; impact depends on neighboring block and data flow.
| 8208 |                                    RichText::new(format!("{}  Set:{:.0}km/h", acc_s, acc_spd)) | support logi
| 8209 |                                        .size(11.0) | support logic; impact depends on neighboring block and d
| 8210 |                                        .color(ac), | support logic; impact depends on neighboring block and d
| 8211 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8212 |                                if ui.small_button("+5").clicked() { | runtime gate; condition changes can ena
| 8213 |                                    do_acc_up = true; | support logic; impact depends on neighboring block and
| 8214 |                                } | scope delimiter; structural only, but wrong placement changes ownership/li
| 8215 |                                if ui.small_button("-5").clicked() { | runtime gate; condition changes can ena
| 8216 |                                    do_acc_dn = true; | support logic; impact depends on neighboring block and
| 8217 |                                } | scope delimiter; structural only, but wrong placement changes ownership/li
| 8218 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 8219 |                            ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and da
| 8220 |                                ui.label(RichText::new("Headway").size(11.0).color(Color32::GREEN)); | support
| 8221 |                                if ui.small_button("-0.2s").clicked() { | runtime gate; condition changes can
| 8222 |                                    do_hdw_dn = true; | support logic; impact depends on neighboring block and
| 8223 |                                } | scope delimiter; structural only, but wrong placement changes ownership/li
| 8224 |                                if ui.small_button("+0.2s").clicked() { | runtime gate; condition changes can
| 8225 |                                    do_hdw_up = true; | support logic; impact depends on neighboring block and
| 8226 |                                } | scope delimiter; structural only, but wrong placement changes ownership/li
| 8227 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 8228 |                            bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 8229 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 8230 |                                "Set spd", | support logic; impact depends on neighboring block and data flow.
| 8231 |                                acc_spd, | support logic; impact depends on neighboring block and data flow. |
| 8232 |                                150.0, | support logic; impact depends on neighboring block and data flow. |
| 8233 |                                "km/h", | support logic; impact depends on neighboring block and data flow. |
| 8234 |                                120.0, | support logic; impact depends on neighboring block and data flow. |
| 8235 |                                Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and
| 8236 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8237 |                            bar_gauge(ui, "Headway", acc_hdw, 4.0, "s", 120.0, Color32::GREEN); | support logi
| 8238 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8239 |                            // AEB | comment/documentation; changing affects understanding and intent traceabi
| 8240 |                            let ab = match aeb_s { | support logic; impact depends on neighboring block and da
| 8241 |                                FeatureState::Active => Color32::RED, | support logic; impact depends on neigh
| 8242 |                                FeatureState::Ready => Color32::GREEN, | support logic; impact depends on neig
| 8243 |                                _ => Color32::from_gray(80), | support logic; impact depends on neighboring bl
| 8244 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifet
| 8245 |                            ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and da
```

```
| 8246 |                                ui.label(RichText::new("AEB:").size(12.0).color(ab).strong()); | support logic
| 8247 |                                ui.label(RichText::new(format!("{}", aeb_s)).size(11.0).color(ab)); | support
| 8248 |                                if aeb_pc { | runtime gate; condition changes can enable/disable critical paths
| 8249 |                                    ui.label( | support logic; impact depends on neighboring block and data flo
| 8250 |                                        RichText::new("PRE-CHG").size(10.5).color(Color32::YELLOW), | support
| 8251 |                                    ); | support logic; impact depends on neighboring block and data flow. |
| 8252 |                                } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 8253 |                                ui.label( | support logic; impact depends on neighboring block and data flow.
| 8254 |                                    RichText::new(format!("{:.0}%", aeb_br * 100.0)) | support logic; impact de
| 8255 |                                        .size(11.0) | support logic; impact depends on neighboring block and da
| 8256 |                                        .color(Color32::RED), | support logic; impact depends on neighboring bl
| 8257 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8258 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 8259 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8260 |                            // LKA | comment/documentation; changing affects understanding and intent traceabi
| 8261 |                            let lc2 = match lka_s { | support logic; impact depends on neighboring block and da
| 8262 |                                FeatureState::Active => Color32::GREEN, | support logic; impact depends on neig
| 8263 |                                _ => Color32::from_gray(80), | support logic; impact depends on neighboring blo
| 8264 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 8265 |                            ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and dat
| 8266 |                                ui.label(RichText::new("LKA:").size(12.0).color(lc2).strong()); | support logic
| 8267 |                                ui.label(RichText::new(format!("{}", lka_s)).size(11.0).color(lc2)); | support
| 8268 |                                if ui.small_button("Toggle").clicked() { | runtime gate; condition changes can
| 8269 |                                    do_lka = true; | support logic; impact depends on neighboring block and dat
| 8270 |                                } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 8271 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 8272 |                            digital_readout( | support logic; impact depends on neighboring block and data flow
| 8273 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 8274 |                                "Steer cmd", | support logic; impact depends on neighboring block and data flow
| 8275 |                                &format!("{:+.2}°", lka_st), | support logic; impact depends on neighboring blo
| 8276 |                                Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 8277 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8278 |                            if !det_s { | runtime gate; condition changes can enable/disable critical paths. |
| 8279 |                                ui.label( | support logic; impact depends on neighboring block and data flow.
| 8280 |                                    RichText::new("LKA em espera: nao ha faixa confiavel detectada.") | suppor
| 8281 |                                        .size(10.0) | support logic; impact depends on neighboring block and da
| 8282 |                                        .color(Color32::YELLOW), | support logic; impact depends on neighboring
| 8283 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8284 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8285 |                            let tj = match tja_s { | support logic; impact depends on neighboring block and dat
| 8286 |                                FeatureState::Active => Color32::GREEN, | support logic; impact depends on neig
| 8287 |                                _ => Color32::from_gray(80), | support logic; impact depends on neighboring blo
| 8288 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 8289 |                            ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and dat
| 8290 |                                ui.label(RichText::new("TJA:").size(12.0).color(tj).strong()); | support logic
| 8291 |                                ui.label(RichText::new(format!("{}", tja_s)).size(11.0).color(tj)); | support
| 8292 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 8293 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 8294 | (blank) | blank separator; no runtime behavior. |
| 8295 |                    // ■■ Centre: 2D path canvas ■■■
```

```
| 8322 |                                 p.line_segment( | support logic; impact depends on neighboring block and data f
| 8323 |                                     [Pos2::new(rect.left(), y), Pos2::new(rect.right(), y)], | support logic; i
| 8324 |                                     Stroke::new(0.5, Color32::from_gray(30)), | support logic; impact depends o
| 8325 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 8326 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8327 |                             // Scale label | comment/documentation; changing affects understanding and intent f
| 8328 |                             p.text( | support logic; impact depends on neighboring block and data flow. |
| 8329 |                                 Pos2::new(rect.left() + 4.0, rect.bottom() - 14.0), | support logic; impact dep
| 8330 |                                 Align2::LEFT_BOTTOM, | support logic; impact depends on neighboring block and o
| 8331 |                                 "Grid: 10m", | support logic; impact depends on neighboring block and data flow
| 8332 |                                 FontId::proportional(9.0), | support logic; impact depends on neighboring block
| 8333 |                                 Color32::from_gray(80), | support logic; impact depends on neighboring block an
| 8334 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8335 |                             // Ego vehicle | comment/documentation; changing affects understanding and intent f
| 8336 |                             p.rect_filled( | support logic; impact depends on neighboring block and data flow.
| 8337 |                                 Rect::from_center_size(Pos2::new(cx, cy), Vec2::new(8.0, 14.0)), | support logi
| 8338 |                                 2.0, | support logic; impact depends on neighboring block and data flow. |
| 8339 |                                 Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and o
| 8340 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8341 |                             p.text( | support logic; impact depends on neighboring block and data flow. |
| 8342 |                                 Pos2::new(cx, cy - 20.0), | support logic; impact depends on neighboring block
| 8343 |                                 Align2::CENTER_CENTER, | support logic; impact depends on neighboring block and
| 8344 |                                 "■", | support logic; impact depends on neighboring block and data flow. |
| 8345 |                                 FontId::proportional(14.0), | support logic; impact depends on neighboring bloc
| 8346 |                                 Color32::WHITE, | support logic; impact depends on neighboring block and data f
| 8347 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8348 |                             // Planned path (from AD system) | comment/documentation; changing affects understa
| 8349 |                             if !path_v.is_empty() { | runtime gate; condition changes can enable/disable critic
| 8350 |                                 let pts: Vec<Pos2> = path_v | support logic; impact depends on neighboring bloc
| 8351 |                                     .iter() | support logic; impact depends on neighboring block and data flow
| 8352 |                                     .map(\|(x, y)\| { | support logic; impact depends on neighboring block and
| 8353 |                                         Pos2::new(cx + *y as f32 * scale, cy - *x as f32 * scale) | support log
| 8354 |                                     }) | support logic; impact depends on neighboring block and data flow. |
| 8355 |                                     .collect(); | support logic; impact depends on neighboring block and data f
| 8356 |                                 for w in pts.windows(2) { | iteration logic; may alter timing, throughput, and
| 8357 |                                     p.line_segment( | support logic; impact depends on neighboring block and da
| 8358 |                                         [w[0], w[1]], | support logic; impact depends on neighboring block and
| 8359 |                                         Stroke::new(1.5, Color32::from_rgb(0, 200, 100)), | support logic; impa
| 8360 |                                     ); | support logic; impact depends on neighboring block and data flow. |
| 8361 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 8362 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8363 |                             // User waypoints | comment/documentation; changing affects understanding and inter
| 8364 |                             let mut add_wp: Option<[f64; 2]> = None; | support logic; impact depends on neighbo
| 8365 |                             for (i, wp) in self.ad_waypoints.iter().enumerate() { | iteration logic; may alter
| 8366 |                                 let px = cx + wp[1] as f32 * scale; | support logic; impact depends on neighbor
| 8367 |                                 let py = cy - wp[0] as f32 * scale; | support logic; impact depends on neighbor
| 8368 |                                 p.circle_filled( | support logic; impact depends on neighboring block and data
| 8369 |                                     Pos2::new(px, py), | support logic; impact depends on neighboring block and
| 8370 |                                     5.0, | support logic; impact depends on neighboring block and data flow. |
| 8371 |                                     Color32::from_rgb(255, 150, 30), | support logic; impact depends on neighbo
| 8372 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 8373 |                                 p.text( | support logic; impact depends on neighboring block and data flow. |
| 8374 |                                     Pos2::new(px + 6.0, py), | support logic; impact depends on neighboring blo
| 8375 |                                     Align2::LEFT_CENTER, | support logic; impact depends on neighboring block a
| 8376 |                                     format!("W{}", i), | support logic; impact depends on neighboring block and
| 8377 |                                     FontId::proportional(9.0), | support logic; impact depends on neighboring b
| 8378 |                                     Color32::from_rgb(255, 150, 30), | support logic; impact depends on neighbo
| 8379 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 8380 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8381 |                             // Connect waypoints | comment/documentation; changing affects understanding and in
| 8382 |                             let all_pts: Vec<Pos2> = self | support logic; impact depends on neighboring block
| 8383 |                                 .ad_waypoints | support logic; impact depends on neighboring block and data flo
| 8384 |                                 .iter() | support logic; impact depends on neighboring block and data flow. |
| 8385 |                                 .map(\|wp\| { | support logic; impact depends on neighboring block and data flo
| 8386 |                                     Pos2::new(cx + wp[1] as f32 * scale, cy - wp[0] as f32 * scale) | support
| 8387 |                                 }) | support logic; impact depends on neighboring block and data flow. |
| 8388 |                                 .collect(); | support logic; impact depends on neighboring block and data flow
| 8389 |                             for w in all_pts.windows(2) { | iteration logic; may alter timing, throughput, and
| 8390 |                                 p.line_segment( | support logic; impact depends on neighboring block and data f
| 8391 |                                     [w[0], w[1]], | support logic; impact depends on neighboring block and data
| 8392 |                                     Stroke::new(1.5, Color32::from_rgb(255, 100, 0)), | support logic; impact o
| 8393 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 8394 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8395 |                             // Click to add waypoint | comment/documentation; changing affects understanding an
| 8396 |                             if resp.clicked() { | runtime gate; condition changes can enable/disable critical p
| 8397 |                                 if let Some(pos) = resp.interact_pointer_pos() { | runtime gate; condition char
```

```
| 8398 |                                     let world_x = (cy - pos.y) as f64 / scale as f64; | support logic; impact d
| 8399 |                                     let world_y = (pos.x - cx) as f64 / scale as f64; | support logic; impact d
| 8400 |                                     add_wp = Some([world_x, world_y]); | support logic; impact depends on neigh
| 8401 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 8402 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8403 |                             if let Some(wp) = add_wp { | runtime gate; condition changes can enable/disable cri
| 8404 |                                 self.cmds.push(Cmd::AddWaypoint(wp[0], wp[1])); | support logic; impact depends
| 8405 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8406 |                             // Buttons below canvas | comment/documentation; changing affects understanding and
| 8407 |                             ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and dat
| 8408 |                                 if ui.button("■ Clear Path").clicked() { | runtime gate; condition changes can
| 8409 |                                     self.cmds.push(Cmd::ClearWaypoints); | support logic; impact depends on nei
| 8410 |                                 } | scope delimiter; structural only, but wrong placement changes ownership/lif
| 8411 |                                 ui.label( | support logic; impact depends on neighboring block and data flow.
| 8412 |                                     RichText::new(format!("{} waypoints", self.ad_waypoints.len())) | support l
| 8413 |                                         .size(11.0) | support logic; impact depends on neighboring block and da
| 8414 |                                         .color(Color32::from_gray(150)), | support logic; impact depends on nei
| 8415 |                                 ); | support logic; impact depends on neighboring block and data flow. |
| 8416 |                             }); | support logic; impact depends on neighboring block and data flow. |
| 8417 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 8418 | (blank) | blank separator; no runtime behavior. |
| 8419 |                     // ■■ Right: Sensor fusion + safety ■■■■■■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentat
| 8420 |                     let ui = &mut cols[2]; | support logic; impact depends on neighboring block and data flow.
| 8421 |                     egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and dat
| 8422 |                         .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and
| 8423 |                         .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 8424 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8425 |                                 RichText::new("CONTROL OUTPUTS") | support logic; impact depends on neighboring
| 8426 |                                     .size(12.0) | support logic; impact depends on neighboring block and data f
| 8427 |                                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on
| 8428 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8429 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 8430 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8431 |                                 "Throttle", | support logic; impact depends on neighboring block and data flow
| 8432 |                                 thr_c * 100.0, | support logic; impact depends on neighboring block and data f
| 8433 |                                 100.0, | support logic; impact depends on neighboring block and data flow. |
| 8434 |                                 "%", | support logic; impact depends on neighboring block and data flow. |
| 8435 |                                 120.0, | support logic; impact depends on neighboring block and data flow. |
| 8436 |                                 Color32::YELLOW, | support logic; impact depends on neighboring block and data
| 8437 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8438 |                             bar_gauge(ui, "Brake", brk_c * 100.0, 100.0, "%", 120.0, Color32::RED); | support l
| 8439 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8440 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8441 |                                 "Steer", | support logic; impact depends on neighboring block and data flow. |
| 8442 |                                 &format!("{:+.1}°", str_c), | support logic; impact depends on neighboring bloc
| 8443 |                                 Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and d
| 8444 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8445 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8446 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8447 |                                 RichText::new("SENSOR FUSION:") | support logic; impact depends on neighboring
| 8448 |                                     .size(11.0) | support logic; impact depends on neighboring block and data f
| 8449 |                                     .color(Color32::from_gray(140)), | support logic; impact depends on neighbo
| 8450 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8451 |                             let tc = if ttc < 2.0 { | support logic; impact depends on neighboring block and da
| 8452 |                                 Color32::RED | support logic; impact depends on neighboring block and data flow
| 8453 |                             } else if ttc < 4.0 { | support logic; impact depends on neighboring block and data
| 8454 |                                 Color32::YELLOW | support logic; impact depends on neighboring block and data f
| 8455 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 8456 |                                 Color32::GREEN | support logic; impact depends on neighboring block and data fl
| 8457 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 8458 |                             digital_readout(ui, "TTC front", &format!("{:.1}s", ttc.min(99.9)), tc); | support
| 8459 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8460 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8461 |                                 "THW front", | support logic; impact depends on neighboring block and data flow
| 8462 |                                 &format!("{:.1}s", thw.min(99.9)), | support logic; impact depends on neighbori
| 8463 |                                 Color32::LIGHT_BLUE, | support logic; impact depends on neighboring block and d
| 8464 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8465 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8466 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8467 |                                 "Lead dist", | support logic; impact depends on neighboring block and data flow
| 8468 |                                 &format!("{:.1}m", lead.min(999.0)), | support logic; impact depends on neighbo
| 8469 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block a
| 8470 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8471 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8472 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8473 |                                 "Lead spd", | support logic; impact depends on neighboring block and data flow
```

```
| 8474 |                                 &format!("{:.1}m/s", lead_v), | support logic; impact depends on neighboring bl
| 8475 |                                 Color32::from_gray(200), | support logic; impact depends on neighboring block a
| 8476 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8477 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8478 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8479 |                                 "BSM L", | support logic; impact depends on neighboring block and data flow. |
| 8480 |                                 if bsml { "■ OBJECT" } else { "clear  " }, | runtime gate; condition changes c
| 8481 |                                 if bsml { | runtime gate; condition changes can enable/disable critical paths.
| 8482 |                                     Color32::YELLOW | support logic; impact depends on neighboring block and da
| 8483 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 8484 |                                     Color32::GREEN | support logic; impact depends on neighboring block and dat
| 8485 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 8486 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8487 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8488 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8489 |                                 "BSM R", | support logic; impact depends on neighboring block and data flow. |
| 8490 |                                 if bsmr { "■ OBJECT" } else { "clear  " }, | runtime gate; condition changes c
| 8491 |                                 if bsmr { | runtime gate; condition changes can enable/disable critical paths.
| 8492 |                                     Color32::YELLOW | support logic; impact depends on neighboring block and da
| 8493 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 8494 |                                     Color32::GREEN | support logic; impact depends on neighboring block and dat
| 8495 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 8496 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8497 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8498 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8499 |                                 RichText::new("LANE INFO:") | support logic; impact depends on neighboring bloc
| 8500 |                                     .size(11.0) | support logic; impact depends on neighboring block and data f
| 8501 |                                     .color(Color32::from_gray(140)), | support logic; impact depends on neighbo
| 8502 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8503 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8504 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8505 |                                 "Detected", | support logic; impact depends on neighboring block and data flow
| 8506 |                                 if det_s { "YES" } else { "NO" }, | runtime gate; condition changes can enable/
| 8507 |                                 if det_s { Color32::GREEN } else { Color32::RED }, | runtime gate; condition ch
| 8508 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8509 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8510 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8511 |                                 "Conf", | support logic; impact depends on neighboring block and data flow. |
| 8512 |                                 &format!("{:.0}%", conf_s * 100.0), | support logic; impact depends on neighbor
| 8513 |                                 if conf_s > 0.7 { | runtime gate; condition changes can enable/disable critical
| 8514 |                                     Color32::GREEN | support logic; impact depends on neighboring block and dat
| 8515 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 8516 |                                     Color32::YELLOW | support logic; impact depends on neighboring block and da
| 8517 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 8518 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8519 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8520 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8521 |                                 "Offset", | support logic; impact depends on neighboring block and data flow.
| 8522 |                                 &format!("{:+.3}m", off_s), | support logic; impact depends on neighboring bloc
| 8523 |                                 if off_s.abs() > 0.8 { | runtime gate; condition changes can enable/disable cri
| 8524 |                                     Color32::RED | support logic; impact depends on neighboring block and data
| 8525 |                                 } else { | support logic; impact depends on neighboring block and data flow. |
| 8526 |                                     Color32::GREEN | support logic; impact depends on neighboring block and dat
| 8527 |                                 }, | support logic; impact depends on neighboring block and data flow. |
| 8528 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8529 |                             digital_readout( | support logic; impact depends on neighboring block and data flow
| 8530 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8531 |                                 "Width", | support logic; impact depends on neighboring block and data flow. |
| 8532 |                                 &format!("{:.1}m", w_s), | support logic; impact depends on neighboring block a
| 8533 |                                 Color32::from_gray(180), | support logic; impact depends on neighboring block a
| 8534 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8535 |                             digital_readout(ui, "L-mark", &lt_s, Color32::from_gray(180)); | support logic; imp
| 8536 |                             digital_readout(ui, "R-mark", &rt_s, Color32::from_gray(180)); | support logic; imp
| 8537 |                             ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8538 |                             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8539 |                                 RichText::new("DRIVER MONITORING:") | support logic; impact depends on neighbor
| 8540 |                                     .size(11.0) | support logic; impact depends on neighboring block and data f
| 8541 |                                     .color(Color32::from_gray(140)), | support logic; impact depends on neighbo
| 8542 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8543 |                             bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 8544 |                                 ui, | support logic; impact depends on neighboring block and data flow. |
| 8545 |                                 "Drowsy", | support logic; impact depends on neighboring block and data flow.
| 8546 |                                 drows, | support logic; impact depends on neighboring block and data flow. |
| 8547 |                                 100.0, | support logic; impact depends on neighboring block and data flow. |
| 8548 |                                 "%", | support logic; impact depends on neighboring block and data flow. |
| 8549 |                                 120.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 8550 |                                if drows > 70.0 { | runtime gate; condition changes can enable/disable critical
| 8551 |                                    Color32::RED | support logic; impact depends on neighboring block and data
| 8552 |                                } else if drows > 40.0 { | support logic; impact depends on neighboring block a
| 8553 |                                    Color32::YELLOW | support logic; impact depends on neighboring block and da
| 8554 |                                } else { | support logic; impact depends on neighboring block and data flow. |
| 8555 |                                    Color32::GREEN | support logic; impact depends on neighboring block and dat
| 8556 |                                }, | support logic; impact depends on neighboring block and data flow. |
| 8557 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8558 |                            digital_readout( | support logic; impact depends on neighboring block and data flow
| 8559 |                                ui, | support logic; impact depends on neighboring block and data flow. |
| 8560 |                                "Hands off", | support logic; impact depends on neighboring block and data flow
| 8561 |                                &format!("{:.0}s", hands), | support logic; impact depends on neighboring block
| 8562 |                                Color32::from_gray(180), | support logic; impact depends on neighboring block a
| 8563 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8564 |                            if alert { | runtime gate; condition changes can enable/disable critical paths. |
| 8565 |                                ui.label( | support logic; impact depends on neighboring block and data flow.
| 8566 |                                    RichText::new("■ DRIVER ATTENTION!") | support logic; impact depends on ne
| 8567 |                                        .size(12.0) | support logic; impact depends on neighboring block and da
| 8568 |                                        .color(Color32::RED) | support logic; impact depends on neighboring bl
| 8569 |                                        .strong(), | support logic; impact depends on neighboring block and dat
| 8570 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8571 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8572 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow
| 8573 |                            let sc = if safe_s.contains("Normal") { | support logic; impact depends on neighbo
| 8574 |                                Color32::GREEN | support logic; impact depends on neighboring block and data f
| 8575 |                            } else if safe_s.contains("Degraded") { | support logic; impact depends on neighbo
| 8576 |                                Color32::YELLOW | support logic; impact depends on neighboring block and data
| 8577 |                            } else { | support logic; impact depends on neighboring block and data flow. |
| 8578 |                                Color32::RED | support logic; impact depends on neighboring block and data flow
| 8579 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifeti
| 8580 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8581 |                                RichText::new(format!("SAFETY: {}", safe_s)) | support logic; impact depends on
| 8582 |                                    .size(12.0) | support logic; impact depends on neighboring block and data
| 8583 |                                    .color(sc) | support logic; impact depends on neighboring block and data f
| 8584 |                                    .strong(), | support logic; impact depends on neighboring block and data f
| 8585 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8586 |                            if let Some(r) = deg_r { | runtime gate; condition changes can enable/disable crit
| 8587 |                                ui.label( | support logic; impact depends on neighboring block and data flow.
| 8588 |                                    RichText::new(format!("→ {}", r)) | support logic; impact depends on neigh
| 8589 |                                        .size(10.5) | support logic; impact depends on neighboring block and da
| 8590 |                                        .color(Color32::RED), | support logic; impact depends on neighboring b
| 8591 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8592 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 8593 |                        }); | support logic; impact depends on neighboring block and data flow. |
| 8594 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 8595 |                }); | support logic; impact depends on neighboring block and data flow. |
| 8596 | (blank) | blank separator; no runtime behavior. |
| 8597 |         // Deferred mutations | comment/documentation; changing affects understanding and intent traceability.
| 8598 |         if do_engage { | runtime gate; condition changes can enable/disable critical paths. |
| 8599 |             self.cmds | support logic; impact depends on neighboring block and data flow. |
| 8600 |                 .push(if eng { Cmd::DisengageAD } else { Cmd::EngageAD }); | support logic; impact depends on
| 8601 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8602 |         if do_lka { | runtime gate; condition changes can enable/disable critical paths. |
| 8603 |             self.cmds.push(Cmd::ToggleLka); | support logic; impact depends on neighboring block and data flow
| 8604 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8605 |         if do_acc_up { | runtime gate; condition changes can enable/disable critical paths. |
| 8606 |             self.cmds.push(Cmd::AccSpeedSet(acc_spd + 5.0)); | support logic; impact depends on neighboring bl
| 8607 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8608 |         if do_acc_dn { | runtime gate; condition changes can enable/disable critical paths. |
| 8609 |             self.cmds.push(Cmd::AccSpeedSet(acc_spd - 5.0)); | support logic; impact depends on neighboring bl
| 8610 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8611 |         if do_hdw_up { | runtime gate; condition changes can enable/disable critical paths. |
| 8612 |             self.cmds.push(Cmd::AccHeadwaySet(acc_hdw + 0.2)); | support logic; impact depends on neighboring
| 8613 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8614 |         if do_hdw_dn { | runtime gate; condition changes can enable/disable critical paths. |
| 8615 |             self.cmds.push(Cmd::AccHeadwaySet(acc_hdw - 0.2)); | support logic; impact depends on neighboring
| 8616 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8617 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8618 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8619 | (blank) | blank separator; no runtime behavior. |
| 8620 | // ■■■
```

```
| 8626 |         let v_cv2x = self.bench.v2x.cv2x_active; | support logic; impact depends on neighboring block and data
| 8627 |         let v_rng = self.bench.v2x.range_m; | support logic; impact depends on neighboring block and data flow
| 8628 |         let v_loss = self.bench.v2x.packet_loss_pct; | support logic; impact depends on neighboring block and d
| 8629 |         let v_lat = self.bench.v2x.latency_ms; | support logic; impact depends on neighboring block and data f
| 8630 |         let v_rng_s = self.v2x_range_ema; | support logic; impact depends on neighboring block and data flow.
| 8631 |         let v_loss_s = self.v2x_loss_ema; | support logic; impact depends on neighboring block and data flow.
| 8632 |         let v_lat_s = self.v2x_lat_ema; | support logic; impact depends on neighboring block and data flow. |
| 8633 |         let mut link_score = 100.0; | support logic; impact depends on neighboring block and data flow. |
| 8634 |         link_score -= (v_loss_s * 2.0).clamp(0.0, 60.0); | support logic; impact depends on neighboring block a
| 8635 |         link_score -= ((v_lat_s - 10.0) * 1.6).clamp(0.0, 40.0); | support logic; impact depends on neighboring
| 8636 |         link_score -= ((180.0 - v_rng_s) * 0.15).clamp(0.0, 20.0); | support logic; impact depends on neighbori
| 8637 |         let link_score = link_score.clamp(0.0, 100.0); | support logic; impact depends on neighboring block and
| 8638 |         let v_tx = self.bench.v2x.bsm_tx_count; | support logic; impact depends on neighboring block and data f
| 8639 |         let v_rx = self.bench.v2x.rx_count; | support logic; impact depends on neighboring block and data flow
| 8640 |         let fca = self.bench.v2x.forward_collision_alert; | support logic; impact depends on neighboring block
| 8641 |         let intx = self.bench.v2x.intersection_alert; | support logic; impact depends on neighboring block and
| 8642 |         let emrg = self.bench.v2x.emergency_vehicle_alert; | support logic; impact depends on neighboring bloc
| 8643 |         let haz = self.bench.v2x.road_hazard_alert; | support logic; impact depends on neighboring block and da
| 8644 |         let gps_lat = self.bench.gps.latitude_deg; | support logic; impact depends on neighboring block and dat
| 8645 |         let gps_lon = self.bench.gps.longitude_deg; | support logic; impact depends on neighboring block and da
| 8646 |         let nearby: Vec<_> = self | support logic; impact depends on neighboring block and data flow. |
| 8647 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 8648 |             .v2x | support logic; impact depends on neighboring block and data flow. |
| 8649 |             .nearby_vehicles | support logic; impact depends on neighboring block and data flow. |
| 8650 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 8651 |             .map(\|v\| { | support logic; impact depends on neighboring block and data flow. |
| 8652 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 8653 |                     v.vehicle_id, | support logic; impact depends on neighboring block and data flow. |
| 8654 |                     v.lat_deg, | support logic; impact depends on neighboring block and data flow. |
| 8655 |                     v.lon_deg, | support logic; impact depends on neighboring block and data flow. |
| 8656 |                     v.speed_ms, | support logic; impact depends on neighboring block and data flow. |
| 8657 |                     v.heading_deg, | support logic; impact depends on neighboring block and data flow. |
| 8658 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 8659 |             }) | support logic; impact depends on neighboring block and data flow. |
| 8660 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 8661 |         let spat: Vec<_> = self | support logic; impact depends on neighboring block and data flow. |
| 8662 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 8663 |             .v2x | support logic; impact depends on neighboring block and data flow. |
| 8664 |             .spat_messages | support logic; impact depends on neighboring block and data flow. |
| 8665 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 8666 |             .map(\|s\| { | support logic; impact depends on neighboring block and data flow. |
| 8667 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 8668 |                     s.intersection_id, | support logic; impact depends on neighboring block and data flow. |
| 8669 |                     s.phase_state, | support logic; impact depends on neighboring block and data flow. |
| 8670 |                     s.time_to_change_s, | support logic; impact depends on neighboring block and data flow. |
| 8671 |                     s.distance_m, | support logic; impact depends on neighboring block and data flow. |
| 8672 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 8673 |             }) | support logic; impact depends on neighboring block and data flow. |
| 8674 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 8675 |         let wz: Vec<_> = self | support logic; impact depends on neighboring block and data flow. |
| 8676 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 8677 |             .v2x | support logic; impact depends on neighboring block and data flow. |
| 8678 |             .work_zones | support logic; impact depends on neighboring block and data flow. |
| 8679 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 8680 |             .map(\|w\| (w.description.clone(), w.distance_m, w.speed_limit_kmh)) | support logic; impact depen
| 8681 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 8682 |         let tel_conn = self.bench.telematics.connect_state; | support logic; impact depends on neighboring bloc
| 8683 |         let tel_bars = self.bench.telematics.signal_bars; | support logic; impact depends on neighboring block
| 8684 |         let tel_rsrp = self.bench.telematics.rsrp_dbm; | support logic; impact depends on neighboring block and
| 8685 |         let tel_ping = self.bench.telematics.ping_ms; | support logic; impact depends on neighboring block and
| 8686 |         let tel_dl = self.bench.telematics.download_kbps; | support logic; impact depends on neighboring block
| 8687 |         let tel_ul = self.bench.telematics.upload_kbps; | support logic; impact depends on neighboring block a
| 8688 |         let tel_data = self.bench.telematics.data_used_mb; | support logic; impact depends on neighboring bloc
| 8689 |         let tel_srv = self.bench.telematics.fleet_server.clone(); | support logic; impact depends on neighbori
| 8690 |         let tel_vid = self.bench.telematics.vehicle_id.clone(); | support logic; impact depends on neighboring
| 8691 |         let tel_sync = self.bench.telematics.last_sync_s; | support logic; impact depends on neighboring block
| 8692 |         let tel_gf = self.bench.telematics.inside_geofence; | support logic; impact depends on neighboring bloc
| 8693 |         let tel_ota_a = self.bench.telematics.ota_available; | support logic; impact depends on neighboring bl
| 8694 |         let tel_ota_d = self.bench.telematics.ota_downloading; | support logic; impact depends on neighboring b
| 8695 |         let tel_ota_v = self.bench.telematics.ota_version.clone(); | support logic; impact depends on neighbori
| 8696 |         let tel_ota_mb = self.bench.telematics.ota_size_mb; | support logic; impact depends on neighboring bloc
| 8697 |         let tel_ota_p = self.bench.telematics.ota_progress_pct; | support logic; impact depends on neighboring
| 8698 |         let tel_cmds: Vec<_> = self | support logic; impact depends on neighboring block and data flow. |
| 8699 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 8700 |             .telematics | support logic; impact depends on neighboring block and data flow. |
| 8701 |             .pending_commands | support logic; impact depends on neighboring block and data flow. |
```

```
| 8702 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 8703 |             .map(\|c\| (format!("{}", c.cmd_type), c.id, c.params.clone())) | support logic; impact depends on
| 8704 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 8705 |         let tel_evs: Vec<_> = self | support logic; impact depends on neighboring block and data flow. |
| 8706 |             .bench | support logic; impact depends on neighboring block and data flow. |
| 8707 |             .telematics | support logic; impact depends on neighboring block and data flow. |
| 8708 |             .events | support logic; impact depends on neighboring block and data flow. |
| 8709 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 8710 |             .map(\|e\| { | support logic; impact depends on neighboring block and data flow. |
| 8711 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 8712 |                     e.timestamp, | support logic; impact depends on neighboring block and data flow. |
| 8713 |                     e.ack, | support logic; impact depends on neighboring block and data flow. |
| 8714 |                     e.sent, | support logic; impact depends on neighboring block and data flow. |
| 8715 |                     format!("{:?}", e.event_type), | error propagation point; changes alter failure propagatio
| 8716 |                     e.description[..e.description.len().min(60)].to_string(), | support logic; impact depends o
| 8717 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 8718 |             }) | support logic; impact depends on neighboring block and data flow. |
| 8719 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 8720 |         let tel_total = self.bench.telematics.total_events_sent; | support logic; impact depends on neighboring
| 8721 |         let elapsed = self.bench.elapsed; | support logic; impact depends on neighboring block and data flow.
| 8722 |         let mut do_ota = false; | support logic; impact depends on neighboring block and data flow. |
| 8723 | (blank) | blank separator; no runtime behavior. |
| 8724 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 8725 |             .id_salt("v2x::page_scroll") | support logic; impact depends on neighboring block and data flow. |
| 8726 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 8727 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 8728 |         ui.columns(2, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 8729 |             let ui = &mut cols[0]; | support logic; impact depends on neighboring block and data flow. |
| 8730 |             egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data flow.
| 8731 |                 .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data flo
| 8732 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 8733 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8734 |                         RichText::new("■ V2X — DSRC 5.9GHz + C-V2X") | support logic; impact depends on neighb
| 8735 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 8736 |                             .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighbo
| 8737 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 8738 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 8739 |                         let dc = if v_dsrc { | support logic; impact depends on neighboring block and data flow
| 8740 |                             Color32::GREEN | support logic; impact depends on neighboring block and data flow.
| 8741 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 8742 |                             Color32::from_gray(60) | support logic; impact depends on neighboring block and dat
| 8743 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 8744 |                         let cc = if v_cv2x { | support logic; impact depends on neighboring block and data flow
| 8745 |                             Color32::GREEN | support logic; impact depends on neighboring block and data flow.
| 8746 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 8747 |                             Color32::from_gray(60) | support logic; impact depends on neighboring block and dat
| 8748 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 8749 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8750 |                             RichText::new(if v_dsrc { "DSRC:ON" } else { "DSRC:off" }) | support logic; impact
| 8751 |                                 .size(11.0) | support logic; impact depends on neighboring block and data flow
| 8752 |                                 .color(dc), | support logic; impact depends on neighboring block and data flow
| 8753 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 8754 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8755 |                             RichText::new(if v_cv2x { "C-V2X:ON" } else { "C-V2X:off" }) | support logic; impac
| 8756 |                                 .size(11.0) | support logic; impact depends on neighboring block and data flow
| 8757 |                                 .color(cc), | support logic; impact depends on neighboring block and data flow
| 8758 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 8759 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8760 |                             RichText::new(format!( | support logic; impact depends on neighboring block and dat
| 8761 |                                 "{:.0}m  Loss:{:.0}%  Lat:{:.0}ms", | support logic; impact depends on neighbor
| 8762 |                                 v_rng, v_loss, v_lat | support logic; impact depends on neighboring block and d
| 8763 |                             )) | support logic; impact depends on neighboring block and data flow. |
| 8764 |                             .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 8765 |                             .color(Color32::from_gray(180)), | support logic; impact depends on neighboring blo
| 8766 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 8767 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 8768 |                     let link_col = if link_score >= 80.0 { | support logic; impact depends on neighboring bloc
| 8769 |                         Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 8770 |                     } else if link_score >= 60.0 { | support logic; impact depends on neighboring block and dat
| 8771 |                         Color32::YELLOW | support logic; impact depends on neighboring block and data flow. |
| 8772 |                     } else { | support logic; impact depends on neighboring block and data flow. |
| 8773 |                         Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 8774 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8775 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8776 |                         RichText::new(format!( | support logic; impact depends on neighboring block and data f
| 8777 |                             "Smoothed link: {:.0}% \| range {:.0}m \| loss {:.1}% \| latency {:.1}ms", | suppo
```

```
| 8778 |                                link_score, v_rng_s, v_loss_s, v_lat_s | support logic; impact depends on neighbori
| 8779 |                            )) | support logic; impact depends on neighboring block and data flow. |
| 8780 |                            .size(10.6) | support logic; impact depends on neighboring block and data flow. |
| 8781 |                            .color(link_col), | support logic; impact depends on neighboring block and data flow.
| 8782 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 8783 |                        digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 8784 |                            ui, | support logic; impact depends on neighboring block and data flow. |
| 8785 |                            "TX/RX", | support logic; impact depends on neighboring block and data flow. |
| 8786 |                            &format!("{}/{}", v_tx, v_rx), | support logic; impact depends on neighboring block and
| 8787 |                            Color32::from_gray(180), | support logic; impact depends on neighboring block and data
| 8788 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 8789 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 8790 |                        for (a, l, c) in [ | iteration logic; may alter timing, throughput, and safety constraints
| 8791 |                            (fca, "■ FORWARD COLLISION", Color32::RED), | support logic; impact depends on neighbo
| 8792 |                            (intx, "■ INTERSECTION RED", Color32::RED), | support logic; impact depends on neighbo
| 8793 |                            (emrg, "■ EMERGENCY VEH", Color32::from_rgb(255, 150, 0)), | support logic; impact dep
| 8794 |                            (haz, "■ ROAD HAZARD", Color32::YELLOW), | support logic; impact depends on neighborin
| 8795 |                        ] { | support logic; impact depends on neighboring block and data flow. |
| 8796 |                            if a { | runtime gate; condition changes can enable/disable critical paths. |
| 8797 |                                ui.label(RichText::new(l).size(12.0).color(c).strong()); | support logic; impact de
| 8798 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 8799 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8800 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 8801 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8802 |                            RichText::new("V2V NEARBY:") | support logic; impact depends on neighboring block and
| 8803 |                                .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 8804 |                                .color(Color32::from_gray(130)), | support logic; impact depends on neighboring bl
| 8805 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 8806 |                        ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow.
| 8807 |                            .id_salt("v2x::nearby_scroll") | support logic; impact depends on neighboring block an
| 8808 |                            .max_height(130.0) | support logic; impact depends on neighboring block and data flow.
| 8809 |                            .auto_shrink([false, false]) | support logic; impact depends on neighboring block and
| 8810 |                            .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 8811 |                                for (vid, vlat, vlon, vspd, vhdg) in &nearby { | iteration logic; may alter timing,
| 8812 |                                    let d = ((*vlat - gps_lat).powi(2) + (*vlon - gps_lon).powi(2)) | support logi
| 8813 |                                        .sqrt() | support logic; impact depends on neighboring block and data flow
| 8814 |                                        * 111320.0; | support logic; impact depends on neighboring block and data
| 8815 |                                    ui.label( | support logic; impact depends on neighboring block and data flow.
| 8816 |                                        RichText::new(format!( | support logic; impact depends on neighboring bloc
| 8817 |                                            "ID:{:04X} {:5.0}m {:.1}m/s Hdg:{:.0}°", | support logic; impact depen
| 8818 |                                            vid, d, vspd, vhdg | support logic; impact depends on neighboring bloc
| 8819 |                                        )) | support logic; impact depends on neighboring block and data flow. |
| 8820 |                                        .size(10.0) | support logic; impact depends on neighboring block and data
| 8821 |                                        .monospace() | support logic; impact depends on neighboring block and data
| 8822 |                                        .color(Color32::from_gray(190)), | support logic; impact depends on neighb
| 8823 |                                    ); | support logic; impact depends on neighboring block and data flow. |
| 8824 |                                } | scope delimiter; structural only, but wrong placement changes ownership/lifetin
| 8825 |                            }); | support logic; impact depends on neighboring block and data flow. |
| 8826 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 8827 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8828 |                            RichText::new("SPaT SIGNALS:") | support logic; impact depends on neighboring block an
| 8829 |                                .size(10.5) | support logic; impact depends on neighboring block and data flow. |
| 8830 |                                .color(Color32::from_gray(130)), | support logic; impact depends on neighboring bl
| 8831 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 8832 |                        for (id, ph, ttc, dist) in &spat { | iteration logic; may alter timing, throughput, and sa
| 8833 |                            let c = match ph { | support logic; impact depends on neighboring block and data flow.
| 8834 |                                auto_breaking::v2x_telematics::TrafficPhase::Green => Color32::GREEN, | support lo
| 8835 |                                auto_breaking::v2x_telematics::TrafficPhase::Yellow => Color32::YELLOW, | support
| 8836 |                                auto_breaking::v2x_telematics::TrafficPhase::Red => Color32::RED, | support logic;
| 8837 |                                _ => Color32::GRAY, | support logic; impact depends on neighboring block and data
| 8838 |                            }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 8839 |                            ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data f
| 8840 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8841 |                                    RichText::new(format!("{:04X}", id)) | support logic; impact depends on neighb
| 8842 |                                        .size(11.0) | support logic; impact depends on neighboring block and data
| 8843 |                                        .color(Color32::from_gray(180)), | support logic; impact depends on neighb
| 8844 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8845 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8846 |                                    RichText::new(format!("{}", ph)) | support logic; impact depends on neighborin
| 8847 |                                        .size(13.0) | support logic; impact depends on neighboring block and data
| 8848 |                                        .color(c) | support logic; impact depends on neighboring block and data fl
| 8849 |                                        .strong(), | support logic; impact depends on neighboring block and data f
| 8850 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8851 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8852 |                                    RichText::new(format!("{:.1}s  {:.0}m", ttc, dist)) | support logic; impact de
| 8853 |                                        .size(11.0) | support logic; impact depends on neighboring block and data
```

```
| 8854 |                                     .color(Color32::from_gray(180)), | support logic; impact depends on neighbo
| 8855 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 8856 |                         }); | support logic; impact depends on neighboring block and data flow. |
| 8857 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8858 |                     ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 8859 |                     for (d, wd, spd) in &wz { | iteration logic; may alter timing, throughput, and safety const
| 8860 |                         let c = if *wd < 100.0 { | support logic; impact depends on neighboring block and data
| 8861 |                             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 8862 |                         } else if *wd < 300.0 { | support logic; impact depends on neighboring block and data f
| 8863 |                             Color32::YELLOW | support logic; impact depends on neighboring block and data flow
| 8864 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 8865 |                             Color32::from_gray(180) | support logic; impact depends on neighboring block and da
| 8866 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 8867 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8868 |                             RichText::new(format!("{} — {:.0}m — {:.0}km/h", d, wd, spd)) | support logic; impa
| 8869 |                                 .size(10.5) | support logic; impact depends on neighboring block and data flow
| 8870 |                                 .color(c), | support logic; impact depends on neighboring block and data flow.
| 8871 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 8872 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8873 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 8874 | (blank) | blank separator; no runtime behavior. |
| 8875 |             let ui = &mut cols[1]; | support logic; impact depends on neighboring block and data flow. |
| 8876 |             egui::Frame::group(ui.style()) | support logic; impact depends on neighboring block and data flow.
| 8877 |                 .fill(Color32::from_gray(15)) | support logic; impact depends on neighboring block and data flo
| 8878 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 8879 |                     ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8880 |                         RichText::new("■ TELEMATICS") | support logic; impact depends on neighboring block and
| 8881 |                             .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 8882 |                             .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighbo
| 8883 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 8884 |                     let cc = match tel_conn { | support logic; impact depends on neighboring block and data flo
| 8885 |                         ConnectState::Connected5G => Color32::GREEN, | support logic; impact depends on neighbo
| 8886 |                         ConnectState::Connected4G => Color32::from_rgb(100, 220, 100), | support logic; impact
| 8887 |                         ConnectState::Connecting => Color32::YELLOW, | support logic; impact depends on neighbo
| 8888 |                         ConnectState::Offline => Color32::RED, | support logic; impact depends on neighboring b
| 8889 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8890 |                     ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 8891 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8892 |                             RichText::new(format!("{}", tel_conn)) | support logic; impact depends on neighbori
| 8893 |                                 .size(13.0) | support logic; impact depends on neighboring block and data flow
| 8894 |                                 .color(cc) | support logic; impact depends on neighboring block and data flow.
| 8895 |                                 .strong(), | support logic; impact depends on neighboring block and data flow.
| 8896 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 8897 |                         for _ in 0..tel_bars { | iteration logic; may alter timing, throughput, and safety cons
| 8898 |                             ui.label(RichText::new("■").size(13.0).color(cc)); | support logic; impact depends
| 8899 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 8900 |                         for _ in tel_bars..5 { | iteration logic; may alter timing, throughput, and safety cons
| 8901 |                             ui.label(RichText::new("■").size(13.0).color(Color32::from_gray(35))); | support l
| 8902 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 8903 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 8904 |                     digital_readout(ui, "Server", &tel_srv, Color32::from_gray(180)); | support logic; impact d
| 8905 |                     digital_readout(ui, "Vehicle", &tel_vid, Color32::YELLOW); | support logic; impact depends
| 8906 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 8907 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 8908 |                         "RSRP", | support logic; impact depends on neighboring block and data flow. |
| 8909 |                         &format!("{:.0} dBm", tel_rsrp), | support logic; impact depends on neighboring block a
| 8910 |                         Color32::from_gray(180), | support logic; impact depends on neighboring block and data
| 8911 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 8912 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 8913 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 8914 |                         "Ping", | support logic; impact depends on neighboring block and data flow. |
| 8915 |                         &format!("{:.0} ms", tel_ping), | support logic; impact depends on neighboring block an
| 8916 |                         Color32::from_gray(180), | support logic; impact depends on neighboring block and data
| 8917 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 8918 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 8919 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 8920 |                         "DL/UL", | support logic; impact depends on neighboring block and data flow. |
| 8921 |                         &format!("{:.0}/{:.0} kbps", tel_dl, tel_ul), | support logic; impact depends on neighb
| 8922 |                         Color32::from_gray(180), | support logic; impact depends on neighboring block and data
| 8923 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 8924 |                     digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 8925 |                         ui, | support logic; impact depends on neighboring block and data flow. |
| 8926 |                         "Data", | support logic; impact depends on neighboring block and data flow. |
| 8927 |                         &format!("{:.2} MB", tel_data), | support logic; impact depends on neighboring block an
| 8928 |                         Color32::from_gray(180), | support logic; impact depends on neighboring block and data
| 8929 |                     ); | support logic; impact depends on neighboring block and data flow. |
```

```
| 8930 |                        digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 8931 |                            ui, | support logic; impact depends on neighboring block and data flow. |
| 8932 |                            "Last sync", | support logic; impact depends on neighboring block and data flow. |
| 8933 |                            &format!("{:.0}s ago", elapsed - tel_sync), | support logic; impact depends on neighbor
| 8934 |                            Color32::from_gray(155), | support logic; impact depends on neighboring block and data
| 8935 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 8936 |                        if let Some(gf) = tel_gf { | runtime gate; condition changes can enable/disable critical pa
| 8937 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8938 |                                RichText::new(format!("■ Inside Geofence #{}", gf)) | support logic; impact depend
| 8939 |                                    .size(11.0) | support logic; impact depends on neighboring block and data flow
| 8940 |                                    .color(Color32::GREEN), | support logic; impact depends on neighboring block a
| 8941 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8942 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8943 |                        if tel_ota_a \|\| tel_ota_d { | runtime gate; condition changes can enable/disable critica
| 8944 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 8945 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8946 |                                RichText::new("OTA UPDATE:") | support logic; impact depends on neighboring block a
| 8947 |                                    .size(11.0) | support logic; impact depends on neighboring block and data flow
| 8948 |                                    .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on ne
| 8949 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8950 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8951 |                                RichText::new(format!("{} ({:.1}MB)", tel_ota_v, tel_ota_mb)) | support logic; impa
| 8952 |                                    .size(11.0) | support logic; impact depends on neighboring block and data flow
| 8953 |                                    .color(Color32::YELLOW), | support logic; impact depends on neighboring block a
| 8954 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8955 |                            if tel_ota_d { | runtime gate; condition changes can enable/disable critical paths. |
| 8956 |                                bar_gauge( | support logic; impact depends on neighboring block and data flow. |
| 8957 |                                    ui, | support logic; impact depends on neighboring block and data flow. |
| 8958 |                                    "Progress", | support logic; impact depends on neighboring block and data flow
| 8959 |                                    tel_ota_p, | support logic; impact depends on neighboring block and data flow.
| 8960 |                                    100.0, | support logic; impact depends on neighboring block and data flow. |
| 8961 |                                    "%", | support logic; impact depends on neighboring block and data flow. |
| 8962 |                                    130.0, | support logic; impact depends on neighboring block and data flow. |
| 8963 |                                    Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring
| 8964 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8965 |                            } else if ui | support logic; impact depends on neighboring block and data flow. |
| 8966 |                                .add( | support logic; impact depends on neighboring block and data flow. |
| 8967 |                                    Button::new(RichText::new("■ Install").size(12.0)) | support logic; impact dep
| 8968 |                                        .fill(Color32::from_rgb(25, 70, 130)), | support logic; impact depends on
| 8969 |                                ) | support logic; impact depends on neighboring block and data flow. |
| 8970 |                                .clicked() | support logic; impact depends on neighboring block and data flow. |
| 8971 |                            { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 8972 |                                do_ota = true; | support logic; impact depends on neighboring block and data flow.
| 8973 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 8974 |                            ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 8975 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8976 |                        if !tel_cmds.is_empty() { | runtime gate; condition changes can enable/disable critical pa
| 8977 |                            ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8978 |                                RichText::new("PENDING COMMANDS:") | support logic; impact depends on neighboring
| 8979 |                                    .size(10.5) | support logic; impact depends on neighboring block and data flow
| 8980 |                                    .color(Color32::YELLOW), | support logic; impact depends on neighboring block a
| 8981 |                            ); | support logic; impact depends on neighboring block and data flow. |
| 8982 |                            for (ct, id, p) in &tel_cmds { | iteration logic; may alter timing, throughput, and sa
| 8983 |                                ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8984 |                                    RichText::new(format!("→ {} [{}] {}", ct, id, p)) | support logic; impact depe
| 8985 |                                        .size(10.5) | support logic; impact depends on neighboring block and data
| 8986 |                                        .color(Color32::YELLOW), | support logic; impact depends on neighboring bl
| 8987 |                                ); | support logic; impact depends on neighboring block and data flow. |
| 8988 |                            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 8989 |                        } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8990 |                        ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 8991 |                        ui.label( | support logic; impact depends on neighboring block and data flow. |
| 8992 |                            RichText::new(format!("EVENTS ({}):", tel_total)) | support logic; impact depends on n
| 8993 |                                .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 8994 |                                .color(Color32::from_gray(140)), | support logic; impact depends on neighboring bl
| 8995 |                        ); | support logic; impact depends on neighboring block and data flow. |
| 8996 |                        ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow.
| 8997 |                            .id_salt("v2x::events_scroll") | support logic; impact depends on neighboring block an
| 8998 |                            .max_height(200.0) | support logic; impact depends on neighboring block and data flow.
| 8999 |                            .auto_shrink([false, false]) | support logic; impact depends on neighboring block and
| 9000 |                            .stick_to_bottom(true) | support logic; impact depends on neighboring block and data f
| 9001 |                            .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow.
| 9002 |                                for (ts, ack, sent, et, desc) in &tel_evs { | iteration logic; may alter timing, t
| 9003 |                                    let c = if et.contains("Dtc") { | support logic; impact depends on neighboring
| 9004 |                                        Color32::RED | support logic; impact depends on neighboring block and data
| 9005 |                                    } else if et.contains("Speed") { | support logic; impact depends on neighborin
```

```
| 9006 |                                 Color32::YELLOW | support logic; impact depends on neighboring block and da
| 9007 |                             } else if et.contains("Ota") { | support logic; impact depends on neighboring b
| 9008 |                                 Color32::from_rgb(0, 210, 210) | support logic; impact depends on neighbori
| 9009 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 9010 |                                 Color32::from_gray(190) | support logic; impact depends on neighboring bloc
| 9011 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/li
| 9012 |                             let a = if *ack { | support logic; impact depends on neighboring block and data
| 9013 |                                 "✓" | support logic; impact depends on neighboring block and data flow. |
| 9014 |                             } else if *sent { | support logic; impact depends on neighboring block and data
| 9015 |                                 "→" | support logic; impact depends on neighboring block and data flow. |
| 9016 |                             } else { | support logic; impact depends on neighboring block and data flow. |
| 9017 |                                 "?" | error propagation point; changes alter failure propagation semantics
| 9018 |                             }; | scope delimiter; structural only, but wrong placement changes ownership/li
| 9019 |                             ui.label( | support logic; impact depends on neighboring block and data flow.
| 9020 |                                 RichText::new(format!("{:.0}s [{}] {} {}", ts, a, et, desc)) | support logi
| 9021 |                                     .size(10.0) | support logic; impact depends on neighboring block and da
| 9022 |                                     .monospace() | support logic; impact depends on neighboring block and c
| 9023 |                                     .color(c), | support logic; impact depends on neighboring block and dat
| 9024 |                             ); | support logic; impact depends on neighboring block and data flow. |
| 9025 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 9026 |                     }); | support logic; impact depends on neighboring block and data flow. |
| 9027 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 9028 |         }); | support logic; impact depends on neighboring block and data flow. |
| 9029 |             }); | support logic; impact depends on neighboring block and data flow. |
| 9030 |         if do_ota { | runtime gate; condition changes can enable/disable critical paths. |
| 9031 |             self.cmds.push(Cmd::StartOta); | support logic; impact depends on neighboring block and data flow.
| 9032 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9033 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9034 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9035 | (blank) | blank separator; no runtime behavior. |
| 9036 | // ═══════════════════════════════════════════════════════════════════════════════ | comment
| 9037 | // TAB UDS | comment/documentation; changing affects understanding and intent traceability. |
| 9038 | // ═══════════════════════════════════════════════════════════════════════════════ | comment
| 9039 | impl App { | implementation binding; behavior semantics and trait conformance live here. |
| 9040 |     fn tab_uds(&mut self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flo
| 9041 |         let uds = &self.bench.uds_ecm; | protocol or I/O critical; changes may impact external integration corr
| 9042 |         let sess_s = format!("{}", uds.session); | protocol or I/O critical; changes may impact external integ
| 9043 |         let sec_s = format!("{:?}", uds.security); | error propagation point; changes alter failure propagatio
| 9044 |         let vin_s = uds.vin.clone(); | protocol or I/O critical; changes may impact external integration corre
| 9045 |         let sw_s = uds.sw_version.clone(); | protocol or I/O critical; changes may impact external integration
| 9046 |         let hw_s = uds.hw_version.clone(); | protocol or I/O critical; changes may impact external integration
| 9047 |         let cal_s = uds.cal_id.clone(); | protocol or I/O critical; changes may impact external integration co
| 9048 |         let ser_s = uds.ecu_serial.clone(); | protocol or I/O critical; changes may impact external integratio
| 9049 |         let idle_s = format!("{:.0}", uds.idle_rpm_cal); | protocol or I/O critical; changes may impact externa
| 9050 |         let rated_s = format!("{:.0}", uds.rated_rpm_cal); | protocol or I/O critical; changes may impact exte
| 9051 |         let torq_s = format!("{:.0} Nm", uds.max_torque_cal); | protocol or I/O critical; changes may impact ex
| 9052 |         let dpf_s = format!("{:.0}%", uds.dpf_regen_threshold); | protocol or I/O critical; changes may impact
| 9053 |         let fm_s = ["Standard", "Economy", "Power"][uds.fuel_map_select as usize]; | protocol or I/O critical;
| 9054 |         let uds_events: Vec<_> = uds | protocol or I/O critical; changes may impact external integration corre
| 9055 |             .event_log | support logic; impact depends on neighboring block and data flow. |
| 9056 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 9057 |             .map(\|e\| { | support logic; impact depends on neighboring block and data flow. |
| 9058 |                 ( | support logic; impact depends on neighboring block and data flow. |
| 9059 |                     e.timestamp, | support logic; impact depends on neighboring block and data flow. |
| 9060 |                     e.service, | support logic; impact depends on neighboring block and data flow. |
| 9061 |                     e.detail.clone(), | support logic; impact depends on neighboring block and data flow. |
| 9062 |                     format!("{:?}", e.result), | error propagation point; changes alter failure propagation se
| 9063 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 9064 |             }) | support logic; impact depends on neighboring block and data flow. |
| 9065 |             .collect(); | support logic; impact depends on neighboring block and data flow. |
| 9066 |         let elapsed = self.bench.elapsed; | support logic; impact depends on neighboring block and data flow.
| 9067 |         let mut send_bytes: Option<(Vec<u8>, u8)> = None; | protocol or I/O critical; changes may impact exter
| 9068 |         let mut do_pick_flash = false; | protocol or I/O critical; changes may impact external integration corr
| 9069 |         let mut do_flash = false; | protocol or I/O critical; changes may impact external integration correctne
| 9070 |         let mut do_clean = false; | support logic; impact depends on neighboring block and data flow. |
| 9071 | (blank) | blank separator; no runtime behavior. |
| 9072 |         ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 9073 |             .id_salt("uds::page_scroll") | protocol or I/O critical; changes may impact external integration co
| 9074 |             .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow. |
| 9075 |             .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 9076 |         ui.columns(2, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 9077 |             let ui = &mut cols[0]; | support logic; impact depends on neighboring block and data flow. |
| 9078 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9079 |                 RichText::new("UDS ISO 14229 CONSOLE") | protocol or I/O critical; changes may impact external
| 9080 |                     .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 9081 |                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighboring blo
```

```
| 9082 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9083 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 9084 |             ui.horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 9085 |                 ui.label(RichText::new("SA:").size(11.0).color(Color32::GRAY)); | support logic; impact depends
| 9086 |                 let mut sa_s = format!("0x{:02X}", self.uds_sa); | protocol or I/O critical; changes may impact
| 9087 |                 if ui | runtime gate; condition changes can enable/disable critical paths. |
| 9088 |                     .add_sized([65.0, 20.0], TextEdit::singleline(&mut sa_s)) | support logic; impact depends o
| 9089 |                     .changed() | support logic; impact depends on neighboring block and data flow. |
| 9090 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9091 |                     if let Ok(v) = u8::from_str_radix(sa_s.trim().trim_start_matches("0x"), 16) { | runtime gat
| 9092 |                         self.uds_sa = v; | protocol or I/O critical; changes may impact external integration co
| 9093 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9094 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9095 |             }); | support logic; impact depends on neighboring block and data flow. |
| 9096 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9097 |                 RichText::new("Request (hex):") | support logic; impact depends on neighboring block and data
| 9098 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 9099 |                     .color(Color32::GRAY), | support logic; impact depends on neighboring block and data flow.
| 9100 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9101 |             ui.add_sized( | support logic; impact depends on neighboring block and data flow. |
| 9102 |                 [ui.available_width() - 10.0, 22.0], | support logic; impact depends on neighboring block and o
| 9103 |                 TextEdit::singleline(&mut self.uds_input).font(FontId::monospace(12.0)), | protocol or I/O crit
| 9104 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9105 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9106 |                 RichText::new("Presets:") | support logic; impact depends on neighboring block and data flow.
| 9107 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 9108 |                     .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block and o
| 9109 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9110 |             ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data flow.
| 9111 |                 for (l, h) in [ | iteration logic; may alter timing, throughput, and safety constraints. |
| 9112 |                     ("ExtSess", "10 02"), | support logic; impact depends on neighboring block and data flow.
| 9113 |                     ("Prog", "10 03"), | support logic; impact depends on neighboring block and data flow. |
| 9114 |                     ("Default", "10 01"), | support logic; impact depends on neighboring block and data flow.
| 9115 |                     ("Seed", "27 01"), | support logic; impact depends on neighboring block and data flow. |
| 9116 |                     ("Keep", "3E 00"), | support logic; impact depends on neighboring block and data flow. |
| 9117 |                     ("ClrDTC", "14 FF FF FF"), | support logic; impact depends on neighboring block and data f
| 9118 |                     ("RdDTC", "19 02 FF"), | support logic; impact depends on neighboring block and data flow.
| 9119 |                     ("RdRPM", "22 DD 01"), | support logic; impact depends on neighboring block and data flow.
| 9120 |                     ("RdCool", "22 DD 03"), | support logic; impact depends on neighboring block and data flow
| 9121 |                     ("RdDPF", "22 DD 08"), | support logic; impact depends on neighboring block and data flow.
| 9122 |                     ("RdVIN", "22 F1 90"), | support logic; impact depends on neighboring block and data flow.
| 9123 |                     ("DPFRgn", "31 01 DF 01"), | support logic; impact depends on neighboring block and data f
| 9124 |                 ] { | support logic; impact depends on neighboring block and data flow. |
| 9125 |                     if ui.small_button(l).clicked() { | runtime gate; condition changes can enable/disable crit
| 9126 |                         self.uds_input = h.into(); | protocol or I/O critical; changes may impact external inte
| 9127 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9128 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9129 |             }); | support logic; impact depends on neighboring block and data flow. |
| 9130 |             ui.add_space(4.0); | support logic; impact depends on neighboring block and data flow. |
| 9131 |             if ui | runtime gate; condition changes can enable/disable critical paths. |
| 9132 |                 .add( | support logic; impact depends on neighboring block and data flow. |
| 9133 |                     Button::new(RichText::new("■ SEND").size(12.0)) | support logic; impact depends on neighbo
| 9134 |                         .fill(Color32::from_rgb(25, 70, 130)) | support logic; impact depends on neighboring b
| 9135 |                         .min_size(Vec2::new(120.0, 28.0)), | support logic; impact depends on neighboring bloc
| 9136 |                 ) | support logic; impact depends on neighboring block and data flow. |
| 9137 |                 .clicked() | support logic; impact depends on neighboring block and data flow. |
| 9138 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9139 |                 let bytes: Vec<u8> = self | support logic; impact depends on neighboring block and data flow.
| 9140 |                     .uds_input | protocol or I/O critical; changes may impact external integration correctness
| 9141 |                     .split_whitespace() | support logic; impact depends on neighboring block and data flow. |
| 9142 |                     .filter_map(\|s\| u8::from_str_radix(s, 16).ok()) | support logic; impact depends on neighb
| 9143 |                     .collect(); | support logic; impact depends on neighboring block and data flow. |
| 9144 |                 if !bytes.is_empty() { | runtime gate; condition changes can enable/disable critical paths. |
| 9145 |                     send_bytes = Some((bytes, self.uds_sa)); | protocol or I/O critical; changes may impact ext
| 9146 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9147 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9148 |             if ui.small_button("■ Clear").clicked() { | runtime gate; condition changes can enable/disable cri
| 9149 |                 self.uds_log.clear(); | protocol or I/O critical; changes may impact external integration corre
| 9150 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9151 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 9152 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9153 |                 RichText::new("ECM FLASH / UPLOAD WORKFLOW") | protocol or I/O critical; changes may impact ext
| 9154 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 9155 |                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighboring bl
| 9156 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9157 |             ui.horizontal_wrapped(\|ui\| { | support logic; impact depends on neighboring block and data flow.
```

```
| 9158 |                 if ui.button("Select FW (.bin)").clicked() { | runtime gate; condition changes can enable/disal
| 9159 |                     do_pick_flash = true; | protocol or I/O critical; changes may impact external integration c
| 9160 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9161 |                 if ui | runtime gate; condition changes can enable/disable critical paths. |
| 9162 |                     .add_enabled(!self.uds_flash_path.is_empty(), Button::new("Flash Upload")) | protocol or I/
| 9163 |                     .clicked() | support logic; impact depends on neighboring block and data flow. |
| 9164 |                 { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9165 |                     do_flash = true; | protocol or I/O critical; changes may impact external integration correc
| 9166 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9167 |                 if ui.button("Clean (ClrDTC + ResetAdap)").clicked() { | runtime gate; condition changes can en
| 9168 |                     do_clean = true; | support logic; impact depends on neighboring block and data flow. |
| 9169 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9170 |             }); | support logic; impact depends on neighboring block and data flow. |
| 9171 |             if self.uds_flash_path.is_empty() { | runtime gate; condition changes can enable/disable critical p
| 9172 |                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9173 |                     RichText::new("FW file: none selected") | support logic; impact depends on neighboring bloc
| 9174 |                         .size(10.0) | support logic; impact depends on neighboring block and data flow. |
| 9175 |                         .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block a
| 9176 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 9177 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 9178 |                 ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9179 |                     RichText::new(format!("FW file: {}", self.uds_flash_path)) | protocol or I/O critical; char
| 9180 |                         .size(9.5) | support logic; impact depends on neighboring block and data flow. |
| 9181 |                         .color(Color32::LIGHT_BLUE), | support logic; impact depends on neighboring block and d
| 9182 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 9183 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9184 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 9185 |             ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 9186 |                 .id_salt("uds::txrx_scroll") | protocol or I/O critical; changes may impact external integratio
| 9187 |                 .max_height(200.0) | support logic; impact depends on neighboring block and data flow. |
| 9188 |                 .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow
| 9189 |                 .stick_to_bottom(true) | support logic; impact depends on neighboring block and data flow. |
| 9190 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 9191 |                     for (is_req, line) in &self.uds_log { | iteration logic; may alter timing, throughput, and
| 9192 |                         let c = if line.contains("■") { | support logic; impact depends on neighboring block a
| 9193 |                             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 9194 |                         } else if !is_req { | support logic; impact depends on neighboring block and data flow
| 9195 |                             Color32::GREEN | support logic; impact depends on neighboring block and data flow.
| 9196 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 9197 |                             Color32::from_gray(200) | support logic; impact depends on neighboring block and da
| 9198 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 9199 |                         ui.label(RichText::new(line).size(10.5).monospace().color(c)); | support logic; impact
| 9200 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9201 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 9202 | (blank) | blank separator; no runtime behavior. |
| 9203 |             let ui = &mut cols[1]; | support logic; impact depends on neighboring block and data flow. |
| 9204 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9205 |                 RichText::new("ECM UDS SERVER STATE") | protocol or I/O critical; changes may impact external i
| 9206 |                     .size(12.0) | support logic; impact depends on neighboring block and data flow. |
| 9207 |                     .color(Color32::from_rgb(80, 155, 255)), | support logic; impact depends on neighboring blo
| 9208 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9209 |             let sc = match sess_s.trim() { | support logic; impact depends on neighboring block and data flow.
| 9210 |                 "DEFAULT" => Color32::from_gray(180), | support logic; impact depends on neighboring block and
| 9211 |                 "EXTENDED" => Color32::YELLOW, | support logic; impact depends on neighboring block and data f
| 9212 |                 _ => Color32::RED, | support logic; impact depends on neighboring block and data flow. |
| 9213 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9214 |             digital_readout(ui, "Session", &sess_s, sc); | support logic; impact depends on neighboring block a
| 9215 |             digital_readout( | support logic; impact depends on neighboring block and data flow. |
| 9216 |                 ui, | support logic; impact depends on neighboring block and data flow. |
| 9217 |                 "Security", | support logic; impact depends on neighboring block and data flow. |
| 9218 |                 &sec_s, | support logic; impact depends on neighboring block and data flow. |
| 9219 |                 if sec_s.contains("Locked") { | runtime gate; condition changes can enable/disable critical pa
| 9220 |                     Color32::from_gray(120) | support logic; impact depends on neighboring block and data flow
| 9221 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 9222 |                     Color32::GREEN | support logic; impact depends on neighboring block and data flow. |
| 9223 |                 }, | support logic; impact depends on neighboring block and data flow. |
| 9224 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9225 |             digital_readout(ui, "VIN", &vin_s, Color32::from_rgb(0, 210, 210)); | support logic; impact depends
| 9226 |             digital_readout(ui, "SW", &sw_s, Color32::from_gray(200)); | support logic; impact depends on neigh
| 9227 |             digital_readout(ui, "HW", &hw_s, Color32::from_gray(200)); | support logic; impact depends on neigh
| 9228 |             digital_readout(ui, "Cal ID", &cal_s, Color32::from_gray(200)); | support logic; impact depends on
| 9229 |             digital_readout(ui, "Serial", &ser_s, Color32::from_gray(200)); | support logic; impact depends on
| 9230 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 9231 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9232 |                 RichText::new("CALIBRATION:") | support logic; impact depends on neighboring block and data flo
| 9233 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
```

```
| 9234 |                     .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block and c
| 9235 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9236 |             digital_readout(ui, "Idle RPM", &idle_s, Color32::YELLOW); | support logic; impact depends on neigh
| 9237 |             digital_readout(ui, "Rated RPM", &rated_s, Color32::YELLOW); | support logic; impact depends on nei
| 9238 |             digital_readout(ui, "Max Torq", &torq_s, Color32::YELLOW); | support logic; impact depends on neigh
| 9239 |             digital_readout(ui, "DPF Regen", &dpf_s, Color32::YELLOW); | support logic; impact depends on neigh
| 9240 |             digital_readout(ui, "Fuel Map", fm_s, Color32::YELLOW); | support logic; impact depends on neighbo
| 9241 |             ui.separator(); | support logic; impact depends on neighboring block and data flow. |
| 9242 |             ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9243 |                 RichText::new("UDS LOG:") | protocol or I/O critical; changes may impact external integration c
| 9244 |                     .size(11.0) | support logic; impact depends on neighboring block and data flow. |
| 9245 |                     .color(Color32::from_gray(140)), | support logic; impact depends on neighboring block and c
| 9246 |             ); | support logic; impact depends on neighboring block and data flow. |
| 9247 |             ScrollArea::vertical() | support logic; impact depends on neighboring block and data flow. |
| 9248 |                 .id_salt("uds::event_log_scroll") | protocol or I/O critical; changes may impact external integ
| 9249 |                 .max_height(160.0) | support logic; impact depends on neighboring block and data flow. |
| 9250 |                 .auto_shrink([false, false]) | support logic; impact depends on neighboring block and data flow
| 9251 |                 .stick_to_bottom(true) | support logic; impact depends on neighboring block and data flow. |
| 9252 |                 .show(ui, \|ui\| { | support logic; impact depends on neighboring block and data flow. |
| 9253 |                     for (ts, svc, det, res) in &uds_events { | iteration logic; may alter timing, throughput, a
| 9254 |                         let c = if res.contains("Positive") { | support logic; impact depends on neighboring bl
| 9255 |                             Color32::GREEN | support logic; impact depends on neighboring block and data flow.
| 9256 |                         } else if res.contains("Security") { | support logic; impact depends on neighboring blo
| 9257 |                             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 9258 |                         } else { | support logic; impact depends on neighboring block and data flow. |
| 9259 |                             Color32::YELLOW | support logic; impact depends on neighboring block and data flow
| 9260 |                         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes
| 9261 |                         ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9262 |                             RichText::new(format!("{:.2}s 0x{:02X} {}", ts, svc, det)) | support logic; impact
| 9263 |                                 .size(10.5) | support logic; impact depends on neighboring block and data flow
| 9264 |                                 .monospace() | support logic; impact depends on neighboring block and data flow
| 9265 |                                 .color(c), | support logic; impact depends on neighboring block and data flow.
| 9266 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 9267 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9268 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 9269 |         }); | support logic; impact depends on neighboring block and data flow. |
| 9270 |             }); | support logic; impact depends on neighboring block and data flow. |
| 9271 | (blank) | blank separator; no runtime behavior. |
| 9272 |         // Execute UDS send AFTER column closure released borrows | comment/documentation; changing affects und
| 9273 |         if let Some((bytes, sa)) = send_bytes { | runtime gate; condition changes can enable/disable critical p
| 9274 |             let ts = elapsed; | support logic; impact depends on neighboring block and data flow. |
| 9275 |             let _ = self.uds_send_and_log(sa, bytes, ts); | protocol or I/O critical; changes may impact extern
| 9276 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9277 |         if do_pick_flash { | runtime gate; condition changes can enable/disable critical paths. |
| 9278 |             if let Some(path) = Self::pick_open_path("bin") { | runtime gate; condition changes can enable/disa
| 9279 |                 self.uds_flash_path = path; | protocol or I/O critical; changes may impact external integration
| 9280 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9281 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9282 |         if do_flash { | runtime gate; condition changes can enable/disable critical paths. |
| 9283 |             let ts = elapsed; | support logic; impact depends on neighboring block and data flow. |
| 9284 |             if matches!(self.hw_cfg.mode, io::hw::HwMode::Live) { | runtime gate; condition changes can enable/
| 9285 |                 if self.uds_flash_path.trim().is_empty() { | runtime gate; condition changes can enable/disable
| 9286 |                     self.uds_log | protocol or I/O critical; changes may impact external integration correctnes
| 9287 |                         .push((false, "            flash failed: selecione .bin".into())); | protocol or I/O cr
| 9288 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 9289 |                     match std::fs::read(&self.uds_flash_path) { | branch dispatcher; arm changes alter behavior
| 9290 |                         Ok(payload) => { | support logic; impact depends on neighboring block and data flow. |
| 9291 |                             match io::live_runner::live_flash_ecm_firmware( | branch dispatcher; arm changes al
| 9292 |                                 &self.hw_cfg, | support logic; impact depends on neighboring block and data flo
| 9293 |                                 Some(self.uds_sa), | protocol or I/O critical; changes may impact external inte
| 9294 |                                 &payload, | support logic; impact depends on neighboring block and data flow.
| 9295 |                             ) { | support logic; impact depends on neighboring block and data flow. |
| 9296 |                                 Ok(s) => self.uds_log.push(( | protocol or I/O critical; changes may impact ext
| 9297 |                                     false, | support logic; impact depends on neighboring block and data flow.
| 9298 |                                     format!( | support logic; impact depends on neighboring block and data flow
| 9299 |                                         "            LIVE flash ok: {} bytes / {} blocks / crc32=0x{:08X} / {:
| 9300 |                                         s.bytes_sent, | support logic; impact depends on neighboring block and
| 9301 |                                         s.blocks_sent, | support logic; impact depends on neighboring block and
| 9302 |                                         s.crc32, | support logic; impact depends on neighboring block and data
| 9303 |                                         s.supply_voltage_v, | support logic; impact depends on neighboring bloc
| 9304 |                                         s.vin, | support logic; impact depends on neighboring block and data f
| 9305 |                                         s.sw_version, | support logic; impact depends on neighboring block and
| 9306 |                                         s.hw_version, | support logic; impact depends on neighboring block and
| 9307 |                                         s.calibration_id, | support logic; impact depends on neighboring block
| 9308 |                                         s.ecu_serial, | support logic; impact depends on neighboring block and
| 9309 |                                         s.transport_diagnostics.transfer_data_blocks_acked, | support logic; in
```

```
| 9310 |                                         s.transport_diagnostics.transfer_data_blocks_attempted, | support logic
| 9311 |                                         s.transport_diagnostics.sequence_error_count, | support logic; impact c
| 9312 |                                         s.transport_diagnostics.flowcontrol_timeout_count | support logic; impa
| 9313 |                                     ), | support logic; impact depends on neighboring block and data flow. |
| 9314 |                                 )), | support logic; impact depends on neighboring block and data flow. |
| 9315 |                                 Err(e) => self | support logic; impact depends on neighboring block and data fl
| 9316 |                                     .uds_log | protocol or I/O critical; changes may impact external integratio
| 9317 |                                     .push((false, format!("            LIVE flash failed: {}", e))), | protocol
| 9318 |                             } | scope delimiter; structural only, but wrong placement changes ownership/lifetim
| 9319 |                         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes.
| 9320 |                         Err(e) => self.uds_log.push(( | protocol or I/O critical; changes may impact external i
| 9321 |                             false, | support logic; impact depends on neighboring block and data flow. |
| 9322 |                             format!("            flash failed: leitura firmware: {}", e), | protocol or I/O cri
| 9323 |                         )), | support logic; impact depends on neighboring block and data flow. |
| 9324 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9325 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9326 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 9327 |                 match self.run_uds_flash_pipeline(self.uds_sa, ts) { | branch dispatcher; arm changes alter bel
| 9328 |                     Ok(msg) => self.uds_log.push((false, format!("            {}", msg))), | protocol or I/O cr
| 9329 |                     Err(e) => self | support logic; impact depends on neighboring block and data flow. |
| 9330 |                         .uds_log | protocol or I/O critical; changes may impact external integration correctnes
| 9331 |                         .push((false, format!("            flash failed: {}", e))), | protocol or I/O critical
| 9332 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9333 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9334 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9335 |         if do_clean { | runtime gate; condition changes can enable/disable critical paths. |
| 9336 |             let ts = elapsed; | support logic; impact depends on neighboring block and data flow. |
| 9337 |             if matches!(self.hw_cfg.mode, io::hw::HwMode::Live) { | runtime gate; condition changes can enable/
| 9338 |                 match io::live_runner::live_clean_ecm(&self.hw_cfg, Some(self.uds_sa)) { | branch dispatcher; a
| 9339 |                     Ok(()) => self | support logic; impact depends on neighboring block and data flow. |
| 9340 |                         .uds_log | protocol or I/O critical; changes may impact external integration correctnes
| 9341 |                         .push((false, "            LIVE clean ok".into())), | support logic; impact depends on
| 9342 |                     Err(e) => self | support logic; impact depends on neighboring block and data flow. |
| 9343 |                         .uds_log | protocol or I/O critical; changes may impact external integration correctnes
| 9344 |                         .push((false, format!("            LIVE clean failed: {}", e))), | support logic; impac
| 9345 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9346 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 9347 |                 let _ = self.uds_send_and_log(self.uds_sa, vec![0x14, 0xFF, 0xFF, 0xFF], ts); | protocol or I/O
| 9348 |                 let _ = self.uds_send_and_log(self.uds_sa, vec![0x31, 0x01, 0xDF, 0x04], ts); | protocol or I/O
| 9349 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9350 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9351 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9352 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9353 | (blank) | blank separator; no runtime behavior. |
| 9354 | // ═══════════════════════════════════════════════════════════════════════════════════ | comment
| 9355 | // TAB PLOTS | comment/documentation; changing affects understanding and intent traceability. |
| 9356 | // ═══════════════════════════════════════════════════════════════════════════════════ | comment
| 9357 | impl App { | implementation binding; behavior semantics and trait conformance live here. |
| 9358 |     fn tab_plots(&self, ui: &mut Ui) { | function signature/body; changes affect API behavior and control-flow
| 9359 |         let h = 155.0_f32; | support logic; impact depends on neighboring block and data flow. |
| 9360 |         // Pre-collect data to avoid borrow conflicts inside columns closure | comment/documentation; changing
| 9361 |         let left = [ | support logic; impact depends on neighboring block and data flow. |
| 9362 |             (&self.pl_rpm, "RPM", Color32::GREEN, 3000.0f64), | support logic; impact depends on neighboring b
| 9363 |             ( | support logic; impact depends on neighboring block and data flow. |
| 9364 |                 &self.pl_torque, | support logic; impact depends on neighboring block and data flow. |
| 9365 |                 "TORQUE Nm", | support logic; impact depends on neighboring block and data flow. |
| 9366 |                 Color32::from_rgb(80, 180, 255), | support logic; impact depends on neighboring block and data
| 9367 |                 1200.0, | support logic; impact depends on neighboring block and data flow. |
| 9368 |             ), | support logic; impact depends on neighboring block and data flow. |
| 9369 |             ( | support logic; impact depends on neighboring block and data flow. |
| 9370 |                 &self.pl_coolant, | support logic; impact depends on neighboring block and data flow. |
| 9371 |                 "COOLANT°C", | support logic; impact depends on neighboring block and data flow. |
| 9372 |                 Color32::from_rgb(255, 130, 30), | support logic; impact depends on neighboring block and data
| 9373 |                 130.0, | support logic; impact depends on neighboring block and data flow. |
| 9374 |             ), | support logic; impact depends on neighboring block and data flow. |
| 9375 |             ( | support logic; impact depends on neighboring block and data flow. |
| 9376 |                 &self.pl_boost, | support logic; impact depends on neighboring block and data flow. |
| 9377 |                 "BOOST kPa", | support logic; impact depends on neighboring block and data flow. |
| 9378 |                 Color32::from_rgb(180, 80, 255), | support logic; impact depends on neighboring block and data
| 9379 |                 300.0, | support logic; impact depends on neighboring block and data flow. |
| 9380 |             ), | support logic; impact depends on neighboring block and data flow. |
| 9381 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 9382 |         let right = [ | support logic; impact depends on neighboring block and data flow. |
| 9383 |             (&self.pl_spd, "SPD km/h", Color32::LIGHT_BLUE, 55.0f64), | support logic; impact depends on neighb
| 9384 |             (&self.pl_fuel, "FUEL L/h", Color32::YELLOW, 30.0), | support logic; impact depends on neighboring
| 9385 |             ( | support logic; impact depends on neighboring block and data flow. |
```

```
| 9386 |                &self.pl_dpf, | support logic; impact depends on neighboring block and data flow. |
| 9387 |                "DPF SOOT%", | support logic; impact depends on neighboring block and data flow. |
| 9388 |                Color32::from_rgb(200, 80, 200), | support logic; impact depends on neighboring block and data
| 9389 |                100.0, | support logic; impact depends on neighboring block and data flow. |
| 9390 |            ), | support logic; impact depends on neighboring block and data flow. |
| 9391 |            ( | support logic; impact depends on neighboring block and data flow. |
| 9392 |                &self.pl_hyd, | support logic; impact depends on neighboring block and data flow. |
| 9393 |                "HYD press", | support logic; impact depends on neighboring block and data flow. |
| 9394 |                Color32::from_rgb(0, 210, 210), | support logic; impact depends on neighboring block and data
| 9395 |                350.0, | support logic; impact depends on neighboring block and data flow. |
| 9396 |            ), | support logic; impact depends on neighboring block and data flow. |
| 9397 |        ]; | support logic; impact depends on neighboring block and data flow. |
| 9398 |        // Clone plot data so we can move into closure without borrow issues | comment/documentation; changing
| 9399 |        let ld: Vec<(Vec<[f64; 2]>, &str, Color32, f64)> = left | support logic; impact depends on neighboring
| 9400 |            .iter() | support logic; impact depends on neighboring block and data flow. |
| 9401 |            .map(\|(v, l, c, m)\| ((*v).clone(), *l, *c, *m)) | support logic; impact depends on neighboring b
| 9402 |            .collect(); | support logic; impact depends on neighboring block and data flow. |
| 9403 |        let rd: Vec<(Vec<[f64; 2]>, &str, Color32, f64)> = right | support logic; impact depends on neighborin
| 9404 |            .iter() | support logic; impact depends on neighboring block and data flow. |
| 9405 |            .map(\|(v, l, c, m)\| ((*v).clone(), *l, *c, *m)) | support logic; impact depends on neighboring b
| 9406 |            .collect(); | support logic; impact depends on neighboring block and data flow. |
| 9407 |        ui.columns(2, \|cols\| { | support logic; impact depends on neighboring block and data flow. |
| 9408 |            for (i, (v, lbl, c, y_max)) in ld.iter().enumerate() { | iteration logic; may alter timing, throug
| 9409 |                let last = v.last().map(\|p\| p[1]).unwrap_or(0.0); | support logic; impact depends on neighbo
| 9410 |                cols[0].horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow
| 9411 |                    ui.label(RichText::new(*lbl).size(11.0).color(*c)); | support logic; impact depends on nei
| 9412 |                    ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9413 |                        RichText::new(format!("{:.1}", last)) | support logic; impact depends on neighboring b
| 9414 |                            .size(13.0) | support logic; impact depends on neighboring block and data flow. |
| 9415 |                            .color(*c) | support logic; impact depends on neighboring block and data flow. |
| 9416 |                            .strong() | support logic; impact depends on neighboring block and data flow. |
| 9417 |                            .monospace(), | support logic; impact depends on neighboring block and data flow.
| 9418 |                    ); | support logic; impact depends on neighboring block and data flow. |
| 9419 |                }); | support logic; impact depends on neighboring block and data flow. |
| 9420 |                let pts_c = v.clone(); | support logic; impact depends on neighboring block and data flow. |
| 9421 |                let cc = *c; | support logic; impact depends on neighboring block and data flow. |
| 9422 |                Plot::new(format!("plots::left::{}::{}", i, lbl)) | support logic; impact depends on neighborin
| 9423 |                    .height(h) | support logic; impact depends on neighboring block and data flow. |
| 9424 |                    .include_y(0.0) | support logic; impact depends on neighboring block and data flow. |
| 9425 |                    .include_y(*y_max) | support logic; impact depends on neighboring block and data flow. |
| 9426 |                    .allow_drag(false) | support logic; impact depends on neighboring block and data flow. |
| 9427 |                    .allow_zoom(false) | support logic; impact depends on neighboring block and data flow. |
| 9428 |                    .allow_scroll(false) | support logic; impact depends on neighboring block and data flow. |
| 9429 |                    .show_axes([false, true]) | support logic; impact depends on neighboring block and data flo
| 9430 |                    .show(&mut cols[0], \|pu\| { | support logic; impact depends on neighboring block and data
| 9431 |                        let pts: PlotPoints = pts_c.iter().map(\|p\| [p[0], p[1]]).collect(); | support logic;
| 9432 |                        pu.line(Line::new(pts).color(cc).width(1.8)); | support logic; impact depends on neighb
| 9433 |                    }); | support logic; impact depends on neighboring block and data flow. |
| 9434 |                cols[0].add_space(4.0); | support logic; impact depends on neighboring block and data flow. |
| 9435 |            } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9436 |            for (i, (v, lbl, c, y_max)) in rd.iter().enumerate() { | iteration logic; may alter timing, throug
| 9437 |                let last = v.last().map(\|p\| p[1]).unwrap_or(0.0); | support logic; impact depends on neighbo
| 9438 |                cols[1].horizontal(\|ui\| { | support logic; impact depends on neighboring block and data flow
| 9439 |                    ui.label(RichText::new(*lbl).size(11.0).color(*c)); | support logic; impact depends on nei
| 9440 |                    ui.label( | support logic; impact depends on neighboring block and data flow. |
| 9441 |                        RichText::new(format!("{:.1}", last)) | support logic; impact depends on neighboring b
| 9442 |                            .size(13.0) | support logic; impact depends on neighboring block and data flow. |
| 9443 |                            .color(*c) | support logic; impact depends on neighboring block and data flow. |
| 9444 |                            .strong() | support logic; impact depends on neighboring block and data flow. |
| 9445 |                            .monospace(), | support logic; impact depends on neighboring block and data flow.
| 9446 |                    ); | support logic; impact depends on neighboring block and data flow. |
| 9447 |                }); | support logic; impact depends on neighboring block and data flow. |
| 9448 |                let pts_c = v.clone(); | support logic; impact depends on neighboring block and data flow. |
| 9449 |                let cc = *c; | support logic; impact depends on neighboring block and data flow. |
| 9450 |                Plot::new(format!("plots::right::{}::{}", i, lbl)) | support logic; impact depends on neighbor
| 9451 |                    .height(h) | support logic; impact depends on neighboring block and data flow. |
| 9452 |                    .include_y(0.0) | support logic; impact depends on neighboring block and data flow. |
| 9453 |                    .include_y(*y_max) | support logic; impact depends on neighboring block and data flow. |
| 9454 |                    .allow_drag(false) | support logic; impact depends on neighboring block and data flow. |
| 9455 |                    .allow_zoom(false) | support logic; impact depends on neighboring block and data flow. |
| 9456 |                    .allow_scroll(false) | support logic; impact depends on neighboring block and data flow. |
| 9457 |                    .show_axes([false, true]) | support logic; impact depends on neighboring block and data flo
| 9458 |                    .show(&mut cols[1], \|pu\| { | support logic; impact depends on neighboring block and data
| 9459 |                        let pts: PlotPoints = pts_c.iter().map(\|p\| [p[0], p[1]]).collect(); | support logic;
| 9460 |                        pu.line(Line::new(pts).color(cc).width(1.8)); | support logic; impact depends on neighb
| 9461 |                    }); | support logic; impact depends on neighboring block and data flow. |
```

```
| 9462 |                 cols[1].add_space(4.0); | support logic; impact depends on neighboring block and data flow. |
| 9463 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9464 |         }); | support logic; impact depends on neighboring block and data flow. |
| 9465 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9466 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/network_mgmt.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 340 | Non-empty: 307 | Symbol count: 23

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 11 |
| is_online | 3 |
| fmt | 2 |
| default | 1 |
| tick | 1 |
| push | 1 |
| is_empty | 1 |
| all | 1 |
| request_sleep | 1 |
| wake_all | 1 |
| node_state | 1 |
| active_count | 1 |
| total_nodes | 1 |
| len | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 22 | enum | NmState | data/schema coupling and match/serde break risk |
| 35 | impl | std::fmt::Display for NmState | method behavior and trait semantics |
| 36 | fn | fmt | API and behavior coupling to callers/implementers |
| 50 | enum | NmFrameType | data/schema coupling and match/serde break risk |
| 64 | struct | NmNode | data/schema coupling and match/serde break risk |
| 77 | impl | NmNode | method behavior and trait semantics |
| 78 | fn | new | API and behavior coupling to callers/implementers |
| 90 | fn | is_online | API and behavior coupling to callers/implementers |
| 98 | enum | BusNmState | data/schema coupling and match/serde break risk |
| 111 | impl | std::fmt::Display for BusNmState | method behavior and trait semantics |
| 112 | fn | fmt | API and behavior coupling to callers/implementers |
| 125 | struct | NetworkManager | data/schema coupling and match/serde break risk |
| 146 | struct | NmEvent | data/schema coupling and match/serde break risk |
| 155 | impl | Default for NetworkManager | method behavior and trait semantics |
| 156 | fn | default | API and behavior coupling to callers/implementers |
| 161 | impl | NetworkManager | method behavior and trait semantics |
| 162 | fn | new | API and behavior coupling to callers/implementers |
| 189 | fn | tick | API and behavior coupling to callers/implementers |
| 306 | fn | request_sleep | API and behavior coupling to callers/implementers |
| 316 | fn | wake_all | API and behavior coupling to callers/implementers |
| 329 | fn | node_state | API and behavior coupling to callers/implementers |
| 333 | fn | active_count | API and behavior coupling to callers/implementers |
| 337 | fn | total_nodes | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Network Management — OSEK NM / AUTOSAR NM (simplified) | comment/documentation; changing affects understanding
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! In a real vehicle, Network Management coordinates ECU lifecycle: | comment/documentation; changing affects und
| 4 | //!   • When the bus goes idle (all ECUs stop transmitting), the bus transitions | comment/documentation; changing
| 5 | //!     to "Bus Sleep" to save power. | comment/documentation; changing affects understanding and intent traceabi
| 6 | //!   • Any ECU needing the bus wakes it up by sending NM frames. | comment/documentation; changing affects unders
| 7 | //!   • ECUs that need to communicate request "Network Requested" state. | comment/documentation; changing affects
| 8 | //!   • Coordinated shutdown: every ECU confirms it is ready to sleep before | comment/documentation; changing af
| 9 | //!     the gateway allows the bus to go off. | comment/documentation; changing affects understanding and intent t
| 10 | //! | comment/documentation; changing affects understanding and intent traceability. |
```

```
| 11 | //! OSEK NM (ISO 17356-5) — used in heavy machinery / trucks | comment/documentation; changing affects understand
| 12 | //! AUTOSAR NM (AUTOSAR_SWS_NetworkManagement) — newer passenger cars | comment/documentation; changing affects u
| 13 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 14 | //! State machine per ECU: | comment/documentation; changing affects understanding and intent traceability. |
| 15 | //!   BusSleep → Prepare (wake) → Normal → Ready Sleep → Bus Sleep | comment/documentation; changing affects u
| 16 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 17 | //! PGN used: J1939 Prop-B in this simulation; real OSEK NM uses CAN IDs 0x400-0x7FF | comment/documentation; cha
| 18 | (blank) | blank separator; no runtime behavior. |
| 19 | // ■■ NM Node State ■■■
```

```
| 87 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 88 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 | (blank) | blank separator; no runtime behavior. |
| 90 |     pub fn is_online(&self) -> bool { | function signature/body; changes affect API behavior and control-flow. |
| 91 |         matches!(self.state, NmState::NormalOperation \| NmState::ReadyToSleep) | support logic; impact depends o
| 92 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 93 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 94 | (blank) | blank separator; no runtime behavior. |
| 95 | // ■■ Bus NM State ■■■
```

```
| 163 |         // Register all ECUs that participate in NM | comment/documentation; changing affects understanding and
| 164 |         // In real OSEK NM, each ECU has a "node ID" separate from J1939 SA | comment/documentation; changing a
| 165 |         let nodes = vec![ | support logic; impact depends on neighboring block and data flow. |
| 166 |             NmNode::new(0x00, "ECM #1", false), | support logic; impact depends on neighboring block and data f
| 167 |             NmNode::new(0x03, "TCM", false), | support logic; impact depends on neighboring block and data flow
| 168 |             NmNode::new(0x27, "BCM/CAB", true), // BCM is typically NM coordinator | support logic; impact depe
| 169 |             NmNode::new(0x1C, "ICM/DASH", false), | support logic; impact depends on neighboring block and data
| 170 |             NmNode::new(0x1E, "HCM", false), | support logic; impact depends on neighboring block and data flow
| 171 |             NmNode::new(0x0B, "ABS/ESP", false), | support logic; impact depends on neighboring block and data
| 172 |             NmNode::new(0x26, "ISOBUS-VT", false), | support logic; impact depends on neighboring block and data
| 173 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 174 | (blank) | blank separator; no runtime behavior. |
| 175 |         NetworkManager { | support logic; impact depends on neighboring block and data flow. |
| 176 |             nodes, | support logic; impact depends on neighboring block and data flow. |
| 177 |             bus_state: BusNmState::Off, | support logic; impact depends on neighboring block and data flow. |
| 178 |             node_timeout_s: 0.5, | support logic; impact depends on neighboring block and data flow. |
| 179 |             sleep_countdown_s: 2.0, | support logic; impact depends on neighboring block and data flow. |
| 180 |             sleep_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 181 |             wakeup_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 182 |             elapsed: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 183 |             event_log: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 184 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 185 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 186 | (blank) | blank separator; no runtime behavior. |
| 187 |     // ━━━
```

```
| 239 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 240 | (blank) | blank separator; no runtime behavior. |
| 241 |         // ■ Bus state machine ■■■
```

```
| 315 |     /// Wake the bus (e.g., key-on event) | comment/documentation; changing affects understanding and intent tra
| 316 |     pub fn wake_all(&mut self, reason: &'static str) { | function signature/body; changes affect API behavior an
| 317 |         for node in &mut self.nodes { | iteration logic; may alter timing, throughput, and safety constraints.
| 318 |             if node.state == NmState::BusSleep { | runtime gate; condition changes can enable/disable critical p
| 319 |                 node.state = NmState::Initializing; | support logic; impact depends on neighboring block and dat
| 320 |                 node.last_nm_ts = self.elapsed; | support logic; impact depends on neighboring block and data fl
| 321 |                 node.wakeup_reason = Some(reason); | support logic; impact depends on neighboring block and data
| 322 |                 node.sleep_confirmed = false; | support logic; impact depends on neighboring block and data flow
| 323 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 324 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 325 |         self.bus_state = BusNmState::WakingUp; | support logic; impact depends on neighboring block and data flo
| 326 |         self.wakeup_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 327 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 328 | (blank) | blank separator; no runtime behavior. |
| 329 |     pub fn node_state(&self, sa: u8) -> Option<NmState> { | function signature/body; changes affect API behavior
| 330 |         self.nodes.iter().find(\|n\| n.sa == sa).map(\|n\| n.state) | support logic; impact depends on neighbori
| 331 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 332 | (blank) | blank separator; no runtime behavior. |
| 333 |     pub fn active_count(&self) -> usize { | function signature/body; changes affect API behavior and control-flo
| 334 |         self.nodes.iter().filter(\|n\| n.is_online()).count() | support logic; impact depends on neighboring blo
| 335 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 336 | (blank) | blank separator; no runtime behavior. |
| 337 |     pub fn total_nodes(&self) -> usize { | function signature/body; changes affect API behavior and control-flow
| 338 |         self.nodes.len() | support logic; impact depends on neighboring block and data flow. |
| 339 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 340 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/nvm.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 385 | Non-empty: 354 | Symbol count: 40

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new_f64 | 10 |
| new | 10 |
| new_u64 | 8 |
| new_str | 6 |
| flush | 5 |
| write_u64 | 4 |
| write_f64 | 3 |
| is_worn | 2 |
| default | 2 |
| update_fuel_trim | 2 |
| increment_cycle | 2 |
| noise | 2 |
| hours | 1 |
| log | 1 |
| fmt | 1 |
| write_str | 1 |
| wear_pct | 1 |
| tick | 1 |
| on_key_on | 1 |
| len | 1 |
| on_key_off | 1 |
| read_f64 | 1 |
| read_u64 | 1 |
| read_str | 1 |
| worn_blocks | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 21 | enum | NvmRegion | data/schema coupling and match/serde break risk |
| 36 | impl | std::fmt::Display for NvmRegion | method behavior and trait semantics |
| 37 | fn | fmt | API and behavior coupling to callers/implementers |
| 53 | struct | NvmBlock | data/schema coupling and match/serde break risk |
| 67 | impl | NvmBlock | method behavior and trait semantics |
| 68 | fn | new_f64 | API and behavior coupling to callers/implementers |
| 80 | fn | new_u64 | API and behavior coupling to callers/implementers |
```

```
| 92 | fn | new_str | API and behavior coupling to callers/implementers |
| 104 | fn | write_f64 | API and behavior coupling to callers/implementers |
| 109 | fn | write_u64 | API and behavior coupling to callers/implementers |
| 114 | fn | write_str | API and behavior coupling to callers/implementers |
| 119 | fn | is_worn | API and behavior coupling to callers/implementers |
| 122 | fn | wear_pct | API and behavior coupling to callers/implementers |
| 130 | struct | Adaptations | data/schema coupling and match/serde break risk |
| 149 | impl | Default for Adaptations | method behavior and trait semantics |
| 150 | fn | default | API and behavior coupling to callers/implementers |
| 155 | impl | Adaptations | method behavior and trait semantics |
| 156 | fn | new | API and behavior coupling to callers/implementers |
| 170 | fn | update_fuel_trim | API and behavior coupling to callers/implementers |
| 182 | struct | StoredDtcRecord | data/schema coupling and match/serde break risk |
| 195 | impl | StoredDtcRecord | method behavior and trait semantics |
| 196 | fn | new | API and behavior coupling to callers/implementers |
| 209 | fn | increment_cycle | API and behavior coupling to callers/implementers |
| 217 | struct | NvmStore | data/schema coupling and match/serde break risk |
| 233 | impl | Default for NvmStore | method behavior and trait semantics |
| 234 | fn | default | API and behavior coupling to callers/implementers |
| 239 | impl | NvmStore | method behavior and trait semantics |
| 240 | fn | new | API and behavior coupling to callers/implementers |
| 287 | fn | tick | API and behavior coupling to callers/implementers |
| 297 | fn | flush | API and behavior coupling to callers/implementers |
| 309 | fn | on_key_on | API and behavior coupling to callers/implementers |
| 326 | fn | on_key_off | API and behavior coupling to callers/implementers |
| 331 | fn | write_f64 | API and behavior coupling to callers/implementers |
| 336 | fn | write_u64 | API and behavior coupling to callers/implementers |
| 341 | fn | read_f64 | API and behavior coupling to callers/implementers |
| 347 | fn | read_u64 | API and behavior coupling to callers/implementers |
| 353 | fn | read_str | API and behavior coupling to callers/implementers |
| 360 | fn | worn_blocks | API and behavior coupling to callers/implementers |
| 364 | fn | add_stored_dtc | API and behavior coupling to callers/implementers |
| 382 | fn | noise | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! NVM — Non-Volatile Memory simulation (EEPROM / Flash emulation) | comment/documentation; changing affects unde
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! In a real ECU, NVM stores: | comment/documentation; changing affects understanding and intent traceability. |
| 4 | //!   • Odometer and engine hours (survive power cuts) | comment/documentation; changing affects understanding an
| 5 | //!   • Calibration parameters (survives firmware updates) | comment/documentation; changing affects understanding
| 6 | //!   • Diagnostic trouble codes (DTCs persist 40+ drive cycles) | comment/documentation; changing affects unders
| 7 | //!   • Learned/adapted values (fuel trim, injector offsets, gear ratios) | comment/documentation; changing affec
| 8 | //!   • Event counters (number of DPF regens, ABS events, etc.) | comment/documentation; changing affects understa
| 9 | //!   • VIN and ECU identification | comment/documentation; changing affects understanding and intent traceability
| 10 | //!   • Boot counter and last-reset reason | comment/documentation; changing affects understanding and intent tra
| 11 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 12 | //! KAM (Keep Alive Memory): A subset of NVM powered by battery even with IGN off. | comment/documentation; chang
| 13 | //!   Loses content only if battery is disconnected. | comment/documentation; changing affects understanding and
| 14 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 15 | //! Wear levelling: Real NVM has limited write cycles (~100K for EEPROM). | comment/documentation; changing affe
| 16 | //!   This simulation tracks write counts per region. | comment/documentation; changing affects understanding an
| 17 | (blank) | blank separator; no runtime behavior. |
| 18 | // ■■ NVM Region tags ■■■
```

```
| 39 |             NvmRegion::Counters => "COUNTERS   ", | support logic; impact depends on neighboring block and data f
| 40 |             NvmRegion::Calibration => "CALIBRATION", | support logic; impact depends on neighboring block and dat
| 41 |             NvmRegion::FaultStorage => "FAULT_STOR ", | support logic; impact depends on neighboring block and da
| 42 |             NvmRegion::Adaptations => "ADAPTATIONS", | support logic; impact depends on neighboring block and dat
| 43 |             NvmRegion::Identification => "IDENT      ", | support logic; impact depends on neighboring block and
| 44 |             NvmRegion::EventLog => "EVENT_LOG  ", | support logic; impact depends on neighboring block and data f
| 45 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 46 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 47 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 | // ■■ NVM Block ■■■
```

```
| 115 |         self.value_str = v; | support logic; impact depends on neighboring block and data flow. |
| 116 |         self.write_count += 1; | protocol or I/O critical; changes may impact external integration correctness.
| 117 |         self.dirty = true; | support logic; impact depends on neighboring block and data flow. |
| 118 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 119 |     pub fn is_worn(&self) -> bool { | function signature/body; changes affect API behavior and control-flow. |
| 120 |         self.write_count >= self.max_writes | protocol or I/O critical; changes may impact external integration
| 121 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 122 |     pub fn wear_pct(&self) -> f64 { | function signature/body; changes affect API behavior and control-flow. |
| 123 |         self.write_count as f64 / self.max_writes as f64 * 100.0 | protocol or I/O critical; changes may impact
| 124 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 125 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 126 | (blank) | blank separator; no runtime behavior. |
| 127 | // ■■ Adaptation Values ■■■
```

```
| 191 |     pub pending: bool, | support logic; impact depends on neighboring block and data flow. |
| 192 |     pub test_passed: bool, | support logic; impact depends on neighboring block and data flow. |
| 193 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 194 | (blank) | blank separator; no runtime behavior. |
| 195 | impl StoredDtcRecord { | implementation binding; behavior semantics and trait conformance live here. |
| 196 |     pub fn new(spn: u32, fmi: u8) -> Self { | function signature/body; changes affect API behavior and control-f
| 197 |         StoredDtcRecord { | support logic; impact depends on neighboring block and data flow. |
| 198 |             spn, | support logic; impact depends on neighboring block and data flow. |
| 199 |             fmi, | support logic; impact depends on neighboring block and data flow. |
| 200 |             occurrence_count: 1, | support logic; impact depends on neighboring block and data flow. |
| 201 |             drive_cycles_ago: 0, | support logic; impact depends on neighboring block and data flow. |
| 202 |             auto_clear_cycle: 40, | support logic; impact depends on neighboring block and data flow. |
| 203 |             confirmed: false, | support logic; impact depends on neighboring block and data flow. |
| 204 |             pending: true, | support logic; impact depends on neighboring block and data flow. |
| 205 |             test_passed: false, | support logic; impact depends on neighboring block and data flow. |
| 206 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 207 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 208 |     /// Called each key-on — increments cycle counter, returns true if should auto-clear | comment/documentatior
| 209 |     pub fn increment_cycle(&mut self) -> bool { | function signature/body; changes affect API behavior and contr
| 210 |         self.drive_cycles_ago += 1; | support logic; impact depends on neighboring block and data flow. |
| 211 |         self.drive_cycles_ago >= self.auto_clear_cycle | support logic; impact depends on neighboring block and
| 212 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 213 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 214 | (blank) | blank separator; no runtime behavior. |
| 215 | // ■■ NVM Store ■■■
```

```
| 267 |                 "ECU20240811001", | support logic; impact depends on neighboring block and data flow. |
| 268 |                 10, | support logic; impact depends on neighboring block and data flow. |
| 269 |             ), | support logic; impact depends on neighboring block and data flow. |
| 270 |             NvmBlock::new_u64(NvmRegion::Identification, "flash_count", 3, 50), | protocol or I/O critical; char
| 271 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 272 | (blank) | blank separator; no runtime behavior. |
| 273 |         NvmStore { | support logic; impact depends on neighboring block and data flow. |
| 274 |             blocks, | support logic; impact depends on neighboring block and data flow. |
| 275 |             adaptations: Adaptations::new(), | support logic; impact depends on neighboring block and data flow
| 276 |             stored_dtcs: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 277 |             drive_cycle_count: 1247, | support logic; impact depends on neighboring block and data flow. |
| 278 |             battery_disconnect_count: 0, | support logic; impact depends on neighboring block and data flow. |
| 279 |             last_flush_ts: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 280 |             flush_interval_s: 60.0, | support logic; impact depends on neighboring block and data flow. |
| 281 |             total_writes: 0, | protocol or I/O critical; changes may impact external integration correctness. |
| 282 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 283 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 284 | (blank) | blank separator; no runtime behavior. |
| 285 |     // ■■■
```

```
| 343 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 344 |             .find(\|b\| b.name == name) | support logic; impact depends on neighboring block and data flow. |
| 345 |             .map(\|b\| b.value_f64) | support logic; impact depends on neighboring block and data flow. |
| 346 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 347 |     pub fn read_u64(&self, name: &str) -> Option<u64> { | function signature/body; changes affect API behavior a
| 348 |         self.blocks | support logic; impact depends on neighboring block and data flow. |
| 349 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 350 |             .find(\|b\| b.name == name) | support logic; impact depends on neighboring block and data flow. |
| 351 |             .map(\|b\| b.value_u64) | support logic; impact depends on neighboring block and data flow. |
| 352 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 353 |     pub fn read_str(&self, name: &str) -> Option<&str> { | function signature/body; changes affect API behavior
| 354 |         self.blocks | support logic; impact depends on neighboring block and data flow. |
| 355 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 356 |             .find(\|b\| b.name == name) | support logic; impact depends on neighboring block and data flow. |
| 357 |             .map(\|b\| b.value_str.as_str()) | support logic; impact depends on neighboring block and data flow
| 358 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 359 | (blank) | blank separator; no runtime behavior. |
| 360 |     pub fn worn_blocks(&self) -> impl Iterator<Item = &NvmBlock> { | function signature/body; changes affect API
| 361 |         self.blocks.iter().filter(\|b\| b.is_worn()) | support logic; impact depends on neighboring block and da
| 362 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 363 | (blank) | blank separator; no runtime behavior. |
| 364 |     pub fn add_stored_dtc(&mut self, spn: u32, fmi: u8) { | function signature/body; changes affect API behavior
| 365 |         if !self | runtime gate; condition changes can enable/disable critical paths. |
| 366 |             .stored_dtcs | support logic; impact depends on neighboring block and data flow. |
| 367 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 368 |             .any(\|d\| d.spn == spn && d.fmi == fmi) | support logic; impact depends on neighboring block and da
| 369 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 370 |             self.stored_dtcs.push(StoredDtcRecord::new(spn, fmi)); | support logic; impact depends on neighborin
| 371 |         } else if let Some(d) = self | support logic; impact depends on neighboring block and data flow. |
| 372 |             .stored_dtcs | support logic; impact depends on neighboring block and data flow. |
| 373 |             .iter_mut() | support logic; impact depends on neighboring block and data flow. |
| 374 |             .find(\|d\| d.spn == spn && d.fmi == fmi) | support logic; impact depends on neighboring block and d
| 375 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 376 |             d.occurrence_count = d.occurrence_count.saturating_add(1); | support logic; impact depends on neighb
| 377 |             d.drive_cycles_ago = 0; | support logic; impact depends on neighboring block and data flow. |
| 378 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 379 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 380 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 381 | (blank) | blank separator; no runtime behavior. |
| 382 | fn noise(seed: f64) -> f64 { | function signature/body; changes affect API behavior and control-flow. |
| 383 |     let x = (seed * 127.1 + 311.7).sin() * 43758.545; | support logic; impact depends on neighboring block and d
| 384 |     x - x.floor() - 0.5 | support logic; impact depends on neighboring block and data flow. |
| 385 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/observability.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 51 | Non-empty: 45 | Symbol count: 7

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| on_step | 1 |
| on_error | 1 |
| on_replay_failure | 1 |
| log_structured | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 4 | struct | StructuredEvent | data/schema coupling and match/serde break risk |
| 14 | struct | SimMetrics | data/schema coupling and match/serde break risk |
| 21 | impl | SimMetrics | method behavior and trait semantics |
| 22 | fn | on_step | API and behavior coupling to callers/implementers |
| 31 | fn | on_error | API and behavior coupling to callers/implementers |
| 35 | fn | on_replay_failure | API and behavior coupling to callers/implementers |
| 40 | fn | log_structured | API and behavior coupling to callers/implementers |
```

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use serde::Serialize; | import wiring; changing path/name can break compilation and module linkage. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | #[derive(Debug, Clone, Serialize)] | comment/documentation; changing affects understanding and intent traceability
| 4 | pub struct StructuredEvent { | data layout contract; field changes can break callers/serde/tests. |
| 5 |     pub timestamp: f64, | support logic; impact depends on neighboring block and data flow. |
| 6 |     pub level: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 7 |     pub module: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 8 |     pub correlation_id: u64, | support logic; impact depends on neighboring block and data flow. |
| 9 |     pub event: &'static str, | support logic; impact depends on neighboring block and data flow. |
| 10 |     pub details: String, | support logic; impact depends on neighboring block and data flow. |
| 11 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | #[derive(Debug, Clone, Default, Serialize)] | comment/documentation; changing affects understanding and intent tr
| 14 | pub struct SimMetrics { | data layout contract; field changes can break callers/serde/tests. |
| 15 |     pub loop_duration_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub steps_completed: u64, | support logic; impact depends on neighboring block and data flow. |
| 17 |     pub error_count: u64, | support logic; impact depends on neighboring block and data flow. |
| 18 |     pub replay_failures: u64, | support logic; impact depends on neighboring block and data flow. |
| 19 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 20 | (blank) | blank separator; no runtime behavior. |
| 21 | impl SimMetrics { | implementation binding; behavior semantics and trait conformance live here. |
| 22 |     pub fn on_step(&mut self, step_duration_ms: f64) { | function signature/body; changes affect API behavior and
| 23 |         self.steps_completed = self.steps_completed.saturating_add(1); | support logic; impact depends on neighbo
| 24 |         self.loop_duration_ms = if self.steps_completed == 1 { | support logic; impact depends on neighboring blo
| 25 |             step_duration_ms | support logic; impact depends on neighboring block and data flow. |
| 26 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 27 |             self.loop_duration_ms * 0.95 + step_duration_ms * 0.05 | support logic; impact depends on neighboring
| 28 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 29 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 30 | (blank) | blank separator; no runtime behavior. |
| 31 |     pub fn on_error(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 32 |         self.error_count = self.error_count.saturating_add(1); | support logic; impact depends on neighboring blo
| 33 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 |     pub fn on_replay_failure(&mut self) { | function signature/body; changes affect API behavior and control-flow
| 36 |         self.replay_failures = self.replay_failures.saturating_add(1); | support logic; impact depends on neighbo
| 37 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 | (blank) | blank separator; no runtime behavior. |
| 40 | pub fn log_structured(ev: &StructuredEvent) { | function signature/body; changes affect API behavior and control
| 41 |     if let Ok(line) = serde_json::to_string(ev) { | runtime gate; condition changes can enable/disable critical p
| 42 |         #[cfg(feature = "advanced_observability")] | comment/documentation; changing affects understanding and in
| 43 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 44 |             tracing::info!(target: "auto_breaking", "{}", line); | support logic; impact depends on neighboring b
| 45 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 46 |         #[cfg(not(feature = "advanced_observability"))] | comment/documentation; changing affects understanding a
| 47 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 |             println!("{}", line); | support logic; impact depends on neighboring block and data flow. |
| 49 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 50 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 51 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/radar.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 352 | Non-empty: 321 | Symbol count: 25

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 13 |
| update | 8 |
| len | 6 |
| push | 3 |
| fmt | 2 |
| kalman_predict | 2 |
| kalman_update | 2 |
| compute_detection | 2 |
| closest_inpath | 2 |
| default | 1 |
```

```
| total_targets | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 9 | enum | RadarPosition | data/schema coupling and match/serde break risk |
| 18 | impl | std::fmt::Display for RadarPosition | method behavior and trait semantics |
| 19 | fn | fmt | API and behavior coupling to callers/implementers |
| 33 | enum | RadarObjectClass | data/schema coupling and match/serde break risk |
| 43 | impl | std::fmt::Display for RadarObjectClass | method behavior and trait semantics |
| 44 | fn | fmt | API and behavior coupling to callers/implementers |
| 59 | struct | RadarTarget | data/schema coupling and match/serde break risk |
| 77 | impl | RadarTarget | method behavior and trait semantics |
| 78 | fn | new | API and behavior coupling to callers/implementers |
| 110 | fn | kalman_predict | API and behavior coupling to callers/implementers |
| 121 | fn | kalman_update | API and behavior coupling to callers/implementers |
| 142 | struct | RadarSensor | data/schema coupling and match/serde break risk |
| 154 | impl | RadarSensor | method behavior and trait semantics |
| 155 | fn | new | API and behavior coupling to callers/implementers |
| 169 | fn | update | API and behavior coupling to callers/implementers |
| 246 | fn | compute_detection | API and behavior coupling to callers/implementers |
| 254 | fn | closest_inpath | API and behavior coupling to callers/implementers |
| 263 | struct | SimTrafficObj | data/schema coupling and match/serde break risk |
| 273 | struct | RadarSuite | data/schema coupling and match/serde break risk |
| 290 | impl | Default for RadarSuite | method behavior and trait semantics |
| 291 | fn | default | API and behavior coupling to callers/implementers |
| 296 | impl | RadarSuite | method behavior and trait semantics |
| 297 | fn | new | API and behavior coupling to callers/implementers |
| 314 | fn | update | API and behavior coupling to callers/implementers |
| 344 | fn | total_targets | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! RADAR — 77GHz Automotive Radar simulation (Bosch SRR2+ / Continental ARS540 style). | comment/documentation; c
| 2 | //! Models: front long-range (250m), rear short-range (80m), corner radars (120m). | comment/documentation; changi
| 3 | //! Implements: target detection, Kalman tracking, Doppler velocity, RCS model. | comment/documentation; changing
| 4 | (blank) | blank separator; no runtime behavior. |
| 5 | use std::f64::consts::PI; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | // ■■ Radar sensor position ■■■
```

```
| 40 |     Unknown, | support logic; impact depends on neighboring block and data flow. |
| 41 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 42 | (blank) | blank separator; no runtime behavior. |
| 43 | impl std::fmt::Display for RadarObjectClass { | implementation binding; behavior semantics and trait conformance
| 44 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 45 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 46 |             RadarObjectClass::Car => write!(f, "Car        "), | protocol or I/O critical; changes may impact ex
| 47 |             RadarObjectClass::Truck => write!(f, "Truck      "), | protocol or I/O critical; changes may impact
| 48 |             RadarObjectClass::Motorcycle => write!(f, "Motorcycle "), | protocol or I/O critical; changes may im
| 49 |             RadarObjectClass::Pedestrian => write!(f, "Pedestrian "), | protocol or I/O critical; changes may im
| 50 |             RadarObjectClass::Bicycle => write!(f, "Bicycle    "), | protocol or I/O critical; changes may impac
| 51 |             RadarObjectClass::Stationary => write!(f, "Stationary "), | protocol or I/O critical; changes may im
| 52 |             RadarObjectClass::Unknown => write!(f, "Unknown    "), | protocol or I/O critical; changes may impac
| 53 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 54 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 56 | (blank) | blank separator; no runtime behavior. |
| 57 | // ■■ Tracked target ■■■
```

```
| 116 |         self.kf_p[1] += 0.5 * dt; | support logic; impact depends on neighboring block and data flow. |
| 117 |         self.kf_p[2] += self.kf_p[3] * dt + 0.1 * dt * dt; | support logic; impact depends on neighboring block
| 118 |         self.kf_p[3] += 0.3 * dt; | support logic; impact depends on neighboring block and data flow. |
| 119 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 120 | (blank) | blank separator; no runtime behavior. |
| 121 |     fn kalman_update(&mut self, z_range: f64, z_rate: f64, z_azimuth: f64) { | function signature/body; changes
| 122 |         let r_cov = 1.0; // measurement noise | support logic; impact depends on neighboring block and data flo
| 123 |         let k0 = self.kf_p[0] / (self.kf_p[0] + r_cov); | support logic; impact depends on neighboring block an
| 124 |         let k1 = self.kf_p[1] / (self.kf_p[1] + r_cov); | support logic; impact depends on neighboring block an
| 125 |         self.kf_x[0] += k0 * (z_range - self.kf_x[0]); | support logic; impact depends on neighboring block and
| 126 |         self.kf_x[1] += k1 * (z_rate - self.kf_x[1]); | support logic; impact depends on neighboring block and
| 127 |         self.kf_p[0] *= 1.0 - k0; | support logic; impact depends on neighboring block and data flow. |
| 128 |         self.kf_p[1] *= 1.0 - k1; | support logic; impact depends on neighboring block and data flow. |
| 129 |         // Update derived quantities | comment/documentation; changing affects understanding and intent traceab
| 130 |         self.range_m = self.kf_x[0].max(0.0); | support logic; impact depends on neighboring block and data flo
| 131 |         self.range_rate_ms = self.kf_x[1]; | support logic; impact depends on neighboring block and data flow.
| 132 |         self.azimuth_deg = z_azimuth; | support logic; impact depends on neighboring block and data flow. |
| 133 |         self.ttc_s = if self.range_rate_ms < -0.1 { | support logic; impact depends on neighboring block and da
| 134 |             -self.range_m / self.range_rate_ms | support logic; impact depends on neighboring block and data fl
| 135 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 136 |             f64::INFINITY | support logic; impact depends on neighboring block and data flow. |
| 137 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 139 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 140 | (blank) | blank separator; no runtime behavior. |
| 141 | // ■■ Single radar sensor ■■■
```

```
| 192 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 193 | (blank) | blank separator; no runtime behavior. |
| 194 |             // Detection probability based on RCS and range | comment/documentation; changing affects understand
| 195 |             let snr_db = obj.rcs_dbsm - 20.0 * (r / 100.0).log10().max(0.0) * 4.0; | support logic; impact depe
| 196 |             if snr_db < 5.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 197 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 198 |             } // Below detection threshold | support logic; impact depends on neighboring block and data flow.
| 199 | (blank) | blank separator; no runtime behavior. |
| 200 |             // Add measurement noise | comment/documentation; changing affects understanding and intent traceab
| 201 |             let r_noisy = r + n(nt * 1.1 + obj.id as f64) * (0.1 + r * 0.005); | support logic; impact depends
| 202 |             let az_noisy = az + n(nt * 2.3 + obj.id as f64) * 0.3; | support logic; impact depends on neighbori
| 203 |             let rr_noisy = rr + n(nt * 3.7 + obj.id as f64) * 0.15; | support logic; impact depends on neighbor
| 204 | (blank) | blank separator; no runtime behavior. |
| 205 |             // Find existing target or create new | comment/documentation; changing affects understanding and i
| 206 |             if let Some(t) = self | runtime gate; condition changes can enable/disable critical paths. |
| 207 |                 .targets | support logic; impact depends on neighboring block and data flow. |
| 208 |                 .iter_mut() | support logic; impact depends on neighboring block and data flow. |
| 209 |                 .find(\|t\| (t.range_m - r).abs() < 5.0 && (t.azimuth_deg - az).abs() < 5.0) | support logic; i
| 210 |             { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 211 |                 t.kalman_update(r_noisy, rr_noisy, az_noisy); | support logic; impact depends on neighboring bl
| 212 |                 t.tracked_cycles += 1; | support logic; impact depends on neighboring block and data flow. |
| 213 |                 t.confidence = (t.confidence + 0.05).min(1.0); | support logic; impact depends on neighboring b
| 214 |                 updated_ids.push(t.id); | support logic; impact depends on neighboring block and data flow. |
| 215 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 216 |                 let id = self.next_id; | support logic; impact depends on neighboring block and data flow. |
| 217 |                 self.next_id = self.next_id.wrapping_add(1); | support logic; impact depends on neighboring blo
| 218 |                 self.targets.push(RadarTarget::new( | support logic; impact depends on neighboring block and da
| 219 |                     id, | support logic; impact depends on neighboring block and data flow. |
| 220 |                     r_noisy, | support logic; impact depends on neighboring block and data flow. |
| 221 |                     az_noisy, | support logic; impact depends on neighboring block and data flow. |
| 222 |                     rr_noisy, | support logic; impact depends on neighboring block and data flow. |
| 223 |                     obj.rcs_dbsm, | support logic; impact depends on neighboring block and data flow. |
| 224 |                 )); | support logic; impact depends on neighboring block and data flow. |
| 225 |                 updated_ids.push(id); | support logic; impact depends on neighboring block and data flow. |
| 226 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 227 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 228 | (blank) | blank separator; no runtime behavior. |
| 229 |         // ■■ Remove stale targets ■■■
```

```
| 268 |     pub rcs_dbsm: f64, | support logic; impact depends on neighboring block and data flow. |
| 269 |     pub object_type: RadarObjectClass, | support logic; impact depends on neighboring block and data flow. |
| 270 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 271 | (blank) | blank separator; no runtime behavior. |
| 272 | // ■■ Full radar suite ■■■
```

```
| 344 |     pub fn total_targets(&self) -> usize { | function signature/body; changes affect API behavior and control-f
| 345 |         self.front_center.targets.len() | support logic; impact depends on neighboring block and data flow. |
| 346 |             + self.rear_center.targets.len() | support logic; impact depends on neighboring block and data flow
| 347 |             + self.front_left.targets.len() | support logic; impact depends on neighboring block and data flow.
| 348 |             + self.front_right.targets.len() | support logic; impact depends on neighboring block and data flow
| 349 |             + self.rear_left.targets.len() | support logic; impact depends on neighboring block and data flow.
| 350 |             + self.rear_right.targets.len() | support logic; impact depends on neighboring block and data flow.
| 351 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 352 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/sensors.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 168 | Non-empty: 150 | Symbol count: 6

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| noise | 6 |
| new | 3 |
| push | 2 |
| update | 1 |
| in_path | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 5 | struct | RadarTarget | data/schema coupling and match/serde break risk |
| 14 | struct | SensorSuite | data/schema coupling and match/serde break risk |
| 48 | impl | SensorSuite | method behavior and trait semantics |
| 49 | fn | new | API and behavior coupling to callers/implementers |
| 74 | fn | update | API and behavior coupling to callers/implementers |
| 165 | fn | noise | API and behavior coupling to callers/implementers |
```

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use crate::vehicle::TrafficObject; | import wiring; changing path/name can break compilation and module linkage.
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | /// A radar return from the forward/rear 77 GHz radar | comment/documentation; changing affects understanding and
| 4 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 5 | pub struct RadarTarget { | data layout contract; field changes can break callers/serde/tests. |
| 6 |     pub distance: f64,       // m | support logic; impact depends on neighboring block and data flow. |
| 7 |     pub relative_speed: f64, // km/h (positive = approaching) | support logic; impact depends on neighboring bloc
| 8 |     pub angle_deg: f64,      // degrees from forward axis | support logic; impact depends on neighboring block an
| 9 |     pub is_in_path: bool, | support logic; impact depends on neighboring block and data flow. |
| 10 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | /// Complete sensor suite (all simulated) | comment/documentation; changing affects understanding and intent tra
| 13 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 14 | pub struct SensorSuite { | data layout contract; field changes can break callers/serde/tests. |
| 15 |     // 77 GHz RADAR | comment/documentation; changing affects understanding and intent traceability. |
| 16 |     pub forward_targets: Vec<RadarTarget>, | support logic; impact depends on neighboring block and data flow. |
| 17 |     pub rear_targets: Vec<RadarTarget>, | support logic; impact depends on neighboring block and data flow. |
| 18 | (blank) | blank separator; no runtime behavior. |
| 19 |     // Mono camera — lane detection | comment/documentation; changing affects understanding and intent traceabil
| 20 |     pub lane_offset: f64,     // m from lane center (+ = right) | support logic; impact depends on neighboring b
| 21 |     pub lane_width: f64,      // m | support logic; impact depends on neighboring block and data flow. |
| 22 |     pub lane_curvature: f64,  // 1/m | support logic; impact depends on neighboring block and data flow. |
| 23 |     pub lane_confidence: f64, // 0-1 | support logic; impact depends on neighboring block and data flow. |
| 24 |     pub lane_departure_left: bool, | support logic; impact depends on neighboring block and data flow. |
| 25 |     pub lane_departure_right: bool, | support logic; impact depends on neighboring block and data flow. |
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 |     // Blind spot monitoring (short-range radar) | comment/documentation; changing affects understanding and int
| 28 |     pub bsm_left: bool, | support logic; impact depends on neighboring block and data flow. |
| 29 |     pub bsm_right: bool, | support logic; impact depends on neighboring block and data flow. |
| 30 | (blank) | blank separator; no runtime behavior. |
| 31 |     // Ultrasonic parking sensors: [FL,FC,FR, BL,BC,BR + 2 corners] | comment/documentation; changing affects un
| 32 |     pub ultrasonic: [f64; 8], | support logic; impact depends on neighboring block and data flow. |
```

```
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 |     // Traffic Sign Recognition (camera-based) | comment/documentation; changing affects understanding and intent
| 35 |     pub speed_limit_kmh: Option<u32>, | support logic; impact depends on neighboring block and data flow. |
| 36 |     pub stop_sign_ahead: bool, | support logic; impact depends on neighboring block and data flow. |
| 37 | (blank) | blank separator; no runtime behavior. |
| 38 |     // IMU | comment/documentation; changing affects understanding and intent traceability. |
| 39 |     pub longitudinal_g: f64, | support logic; impact depends on neighboring block and data flow. |
| 40 |     pub lateral_g: f64, | support logic; impact depends on neighboring block and data flow. |
| 41 |     pub yaw_rate_imu: f64, // deg/s | support logic; impact depends on neighboring block and data flow. |
| 42 | (blank) | blank separator; no runtime behavior. |
| 43 |     // GPS (simplified) | comment/documentation; changing affects understanding and intent traceability. |
| 44 |     pub gps_speed: f64,   // km/h | support logic; impact depends on neighboring block and data flow. |
| 45 |     pub gps_heading: f64, // degrees | support logic; impact depends on neighboring block and data flow. |
| 46 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 47 | (blank) | blank separator; no runtime behavior. |
| 48 | impl SensorSuite { | implementation binding; behavior semantics and trait conformance live here. |
| 49 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 50 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 51 |             forward_targets: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 52 |             rear_targets: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 53 |             lane_offset: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 54 |             lane_width: 3.5, | support logic; impact depends on neighboring block and data flow. |
| 55 |             lane_curvature: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 56 |             lane_confidence: 0.92, | support logic; impact depends on neighboring block and data flow. |
| 57 |             lane_departure_left: false, | support logic; impact depends on neighboring block and data flow. |
| 58 |             lane_departure_right: false, | support logic; impact depends on neighboring block and data flow. |
| 59 |             bsm_left: false, | support logic; impact depends on neighboring block and data flow. |
| 60 |             bsm_right: false, | support logic; impact depends on neighboring block and data flow. |
| 61 |             ultrasonic: [8.0; 8], | support logic; impact depends on neighboring block and data flow. |
| 62 |             speed_limit_kmh: Some(80), | support logic; impact depends on neighboring block and data flow. |
| 63 |             stop_sign_ahead: false, | support logic; impact depends on neighboring block and data flow. |
| 64 |             longitudinal_g: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 65 |             lateral_g: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 66 |             yaw_rate_imu: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 67 |             gps_speed: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 68 |             gps_heading: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 69 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 70 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 | (blank) | blank separator; no runtime behavior. |
| 72 |     /// Update all sensor readings from vehicle state and traffic | comment/documentation; changing affects under
| 73 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent trace
| 74 |     pub fn update( | function signature/body; changes affect API behavior and control-flow. |
| 75 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 76 |         vehicle_speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 77 |         steering_angle: f64, | support logic; impact depends on neighboring block and data flow. |
| 78 |         acceleration: f64, | support logic; impact depends on neighboring block and data flow. |
| 79 |         heading: f64, | support logic; impact depends on neighboring block and data flow. |
| 80 |         traffic: &[TrafficObject], | support logic; impact depends on neighboring block and data flow. |
| 81 |         elapsed: f64, | support logic; impact depends on neighboring block and data flow. |
| 82 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 83 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 84 |         // --- Forward RADAR ------------------------------------------------ | comment/documentation; changing a
| 85 |         self.forward_targets.clear(); | support logic; impact depends on neighboring block and data flow. |
| 86 |         for obj in traffic { | iteration logic; may alter timing, throughput, and safety constraints. |
| 87 |             if obj.distance > 0.0 && obj.distance < 200.0 { | runtime gate; condition changes can enable/disable
| 88 |                 self.forward_targets.push(RadarTarget { | support logic; impact depends on neighboring block and
| 89 |                     distance: obj.distance, | support logic; impact depends on neighboring block and data flow.
| 90 |                     relative_speed: obj.closing_speed, | support logic; impact depends on neighboring block and
| 91 |                     angle_deg: (obj.lateral_offset / obj.distance.max(1.0)) | support logic; impact depends on n
| 92 |                         .atan() | support logic; impact depends on neighboring block and data flow. |
| 93 |                         .to_degrees(), | support logic; impact depends on neighboring block and data flow. |
| 94 |                     is_in_path: obj.in_path(), | support logic; impact depends on neighboring block and data flo
| 95 |                 }); | support logic; impact depends on neighboring block and data flow. |
| 96 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 97 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 98 | (blank) | blank separator; no runtime behavior. |
| 99 |         // --- Rear RADAR -------------------------------------------------- | comment/documentation; changing a
| 100 |         self.rear_targets.clear(); | support logic; impact depends on neighboring block and data flow. |
| 101 |         for obj in traffic { | iteration logic; may alter timing, throughput, and safety constraints. |
| 102 |             if obj.distance < 0.0 && obj.distance > -50.0 { | runtime gate; condition changes can enable/disabl
| 103 |                 self.rear_targets.push(RadarTarget { | support logic; impact depends on neighboring block and d
| 104 |                     distance: -obj.distance, | support logic; impact depends on neighboring block and data flow
| 105 |                     relative_speed: -obj.closing_speed, | support logic; impact depends on neighboring block an
| 106 |                     angle_deg: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 107 |                     is_in_path: obj.lateral_offset.abs() < 1.5, | support logic; impact depends on neighboring
| 108 |                 }); | support logic; impact depends on neighboring block and data flow. |
```

```
| 109 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 110 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 111 | (blank) | blank separator; no runtime behavior. |
| 112 |         // --- Lane detection ---------------------------------------------- | comment/documentation; changing a
| 113 |         // Drift lane_offset based on steering (simplified) | comment/documentation; changing affects understand
| 114 |         self.lane_offset += steering_angle * 0.001 * dt * vehicle_speed / 3.6; | support logic; impact depends o
| 115 |         self.lane_offset = self.lane_offset.clamp(-2.0, 2.0); | support logic; impact depends on neighboring blo
| 116 |         self.lane_confidence = (0.92 + noise(elapsed * 7.3) * 0.04).clamp(0.6, 1.0); | support logic; impact dep
| 117 |         self.lane_departure_left = self.lane_offset < -1.1; | support logic; impact depends on neighboring block
| 118 |         self.lane_departure_right = self.lane_offset > 1.1; | support logic; impact depends on neighboring block
| 119 | (blank) | blank separator; no runtime behavior. |
| 120 |         // --- BSM (side radar, 70m range) --------------------------------- | comment/documentation; changing a
| 121 |         // Simulate periodic vehicle on right side | comment/documentation; changing affects understanding and i
| 122 |         self.bsm_right = (elapsed / 15.0).sin() > 0.7; | support logic; impact depends on neighboring block and
| 123 |         self.bsm_left = (elapsed / 22.0).sin() > 0.85; | support logic; impact depends on neighboring block and
| 124 | (blank) | blank separator; no runtime behavior. |
| 125 |         // --- Ultrasonic (parking) ---------------------------------------- | comment/documentation; changing a
| 126 |         // Front center sensor reacts to nearest forward target | comment/documentation; changing affects unders
| 127 |         if let Some(nearest) = self | runtime gate; condition changes can enable/disable critical paths. |
| 128 |             .forward_targets | support logic; impact depends on neighboring block and data flow. |
| 129 |             .iter() | support logic; impact depends on neighboring block and data flow. |
| 130 |             .filter(\|t\| t.is_in_path) | support logic; impact depends on neighboring block and data flow. |
| 131 |             .min_by(\|a, b\| a.distance.total_cmp(&b.distance)) | support logic; impact depends on neighboring b
| 132 |         { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 133 |             self.ultrasonic[1] = nearest.distance.min(8.0); // front center | support logic; impact depends on n
| 134 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 135 |             self.ultrasonic[1] = 8.0; | support logic; impact depends on neighboring block and data flow. |
| 136 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 137 |         self.ultrasonic[0] = (self.ultrasonic[1] + 0.3 + noise(elapsed * 5.0) * 0.1).min(8.0); | support logic;
| 138 |         self.ultrasonic[2] = (self.ultrasonic[1] + 0.2 + noise(elapsed * 6.0) * 0.1).min(8.0); | support logic;
| 139 |         // Rear static | comment/documentation; changing affects understanding and intent traceability. |
| 140 |         for i in 3..8 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 141 |             self.ultrasonic[i] = | support logic; impact depends on neighboring block and data flow. |
| 142 |                 (5.0 + noise(elapsed * f64::from(i as u8) * 3.1) * 0.5).clamp(0.1, 8.0); | support logic; impact
| 143 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 144 | (blank) | blank separator; no runtime behavior. |
| 145 |         // --- TSR ---------------------------------------------------------- | comment/documentation; changing a
| 146 |         // Change speed limit sign periodically | comment/documentation; changing affects understanding and inte
| 147 |         if ((elapsed / 30.0) as u32).is_multiple_of(2) { | runtime gate; condition changes can enable/disable cr
| 148 |             self.speed_limit_kmh = Some(80); | support logic; impact depends on neighboring block and data flow
| 149 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 150 |             self.speed_limit_kmh = Some(120); | support logic; impact depends on neighboring block and data flow
| 151 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 152 | (blank) | blank separator; no runtime behavior. |
| 153 |         // --- IMU ---------------------------------------------------------- | comment/documentation; changing a
| 154 |         self.longitudinal_g = acceleration / 9.81; | support logic; impact depends on neighboring block and data
| 155 |         self.lateral_g = steering_angle * vehicle_speed / 3.6 / 9.81 * 0.08; | support logic; impact depends on
| 156 |         self.yaw_rate_imu = steering_angle * vehicle_speed / 3.6 / 2.7; | support logic; impact depends on neigh
| 157 | (blank) | blank separator; no runtime behavior. |
| 158 |         // --- GPS ---------------------------------------------------------- | comment/documentation; changing a
| 159 |         self.gps_speed = vehicle_speed + noise(elapsed * 2.1) * 0.3; | support logic; impact depends on neighbor
| 160 |         self.gps_heading = heading; | support logic; impact depends on neighboring block and data flow. |
| 161 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 162 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 163 | (blank) | blank separator; no runtime behavior. |
| 164 | /// Cheap pseudo-random noise in [-1, 1] | comment/documentation; changing affects understanding and intent trac
| 165 | fn noise(seed: f64) -> f64 { | function signature/body; changes affect API behavior and control-flow. |
| 166 |     let x = (seed * 127.1 + 311.7).sin() * 43758.545; | support logic; impact depends on neighboring block and d
| 167 |     x - x.floor() - 0.5 | support logic; impact depends on neighboring block and data flow. |
| 168 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/sim_core.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 71 | Non-empty: 62 | Symbol count: 22

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| len | 8 |
| get | 7 |
| set | 3 |
```

```
| derivative | 3 |
| step | 3 |
| is_empty | 1 |
| push | 1 |
| euler_integrator_advances_state | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 4 | trait | SimState | API and behavior coupling to callers/implementers |
| 5 | fn | len | API and behavior coupling to callers/implementers |
| 6 | fn | is_empty | API and behavior coupling to callers/implementers |
| 9 | fn | get | API and behavior coupling to callers/implementers |
| 10 | fn | set | API and behavior coupling to callers/implementers |
| 13 | trait | DynamicsModel | API and behavior coupling to callers/implementers |
| 14 | fn | derivative | API and behavior coupling to callers/implementers |
| 17 | trait | Integrator | API and behavior coupling to callers/implementers |
| 18 | fn | step | API and behavior coupling to callers/implementers |
| 22 | struct | EulerIntegrator | data/schema coupling and match/serde break risk |
| 24 | impl | Integrator for EulerIntegrator | method behavior and trait semantics |
| 25 | fn | step | API and behavior coupling to callers/implementers |
| 38 | module | tests | namespace and compile-time linkage |
| 41 | struct | VecState | data/schema coupling and match/serde break risk |
| 42 | impl | SimState for VecState | method behavior and trait semantics |
| 43 | fn | len | API and behavior coupling to callers/implementers |
| 46 | fn | get | API and behavior coupling to callers/implementers |
| 49 | fn | set | API and behavior coupling to callers/implementers |
| 54 | struct | UnitDrift | data/schema coupling and match/serde break risk |
| 55 | impl | DynamicsModel for UnitDrift | method behavior and trait semantics |
| 56 | fn | derivative | API and behavior coupling to callers/implementers |
| 62 | fn | euler_integrator_advances_state | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Incremental architecture layer for solver/integrator responsibilities. | comment/documentation; changing affe
| 2 | //! This module is API-preserving: public simulation APIs can migrate internally. | comment/documentation; changi
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | pub trait SimState { | interface contract; changes impact all implementations and users. |
| 5 |     fn len(&self) -> usize; | function signature/body; changes affect API behavior and control-flow. |
| 6 |     fn is_empty(&self) -> bool { | function signature/body; changes affect API behavior and control-flow. |
| 7 |         self.len() == 0 | support logic; impact depends on neighboring block and data flow. |
| 8 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 9 |     fn get(&self, idx: usize) -> f64; | function signature/body; changes affect API behavior and control-flow. |
| 10 |     fn set(&mut self, idx: usize, value: f64); | function signature/body; changes affect API behavior and contro
| 11 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 12 | (blank) | blank separator; no runtime behavior. |
| 13 | pub trait DynamicsModel { | interface contract; changes impact all implementations and users. |
| 14 |     fn derivative(&self, x: &[f64], t: f64) -> Vec<f64>; | function signature/body; changes affect API behavior a
| 15 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 | pub trait Integrator { | interface contract; changes impact all implementations and users. |
| 18 |     fn step(&self, model: &dyn DynamicsModel, state: &mut dyn SimState, t: f64, dt: f64); | function signature/b
| 19 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 20 | (blank) | blank separator; no runtime behavior. |
| 21 | #[derive(Debug, Clone, Copy, Default)] | comment/documentation; changing affects understanding and intent traceal
| 22 | pub struct EulerIntegrator; | data layout contract; field changes can break callers/serde/tests. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | impl Integrator for EulerIntegrator { | implementation binding; behavior semantics and trait conformance live he
| 25 |     fn step(&self, model: &dyn DynamicsModel, state: &mut dyn SimState, t: f64, dt: f64) { | function signature/
| 26 |         let mut x = Vec::with_capacity(state.len()); | support logic; impact depends on neighboring block and da
| 27 |         for i in 0..state.len() { | iteration logic; may alter timing, throughput, and safety constraints. |
| 28 |             x.push(state.get(i)); | support logic; impact depends on neighboring block and data flow. |
| 29 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 30 |         let dx = model.derivative(&x, t); | support logic; impact depends on neighboring block and data flow. |
| 31 |         for (i, dxi) in dx.iter().enumerate().take(state.len()) { | iteration logic; may alter timing, throughpu
| 32 |             state.set(i, state.get(i) + dxi * dt); | support logic; impact depends on neighboring block and data
| 33 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 34 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 35 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 36 | (blank) | blank separator; no runtime behavior. |
| 37 | #[cfg(test)] | comment/documentation; changing affects understanding and intent traceability. |
| 38 | mod tests { | module declaration; changing can break namespace graph. |
```

```
| 39 |     use super::*; | import wiring; changing path/name can break compilation and module linkage. |
| 40 | (blank) | blank separator; no runtime behavior. |
| 41 |     struct VecState(Vec<f64>); | data layout contract; field changes can break callers/serde/tests. |
| 42 |     impl SimState for VecState { | implementation binding; behavior semantics and trait conformance live here. |
| 43 |         fn len(&self) -> usize { | function signature/body; changes affect API behavior and control-flow. |
| 44 |             self.0.len() | support logic; impact depends on neighboring block and data flow. |
| 45 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 46 |         fn get(&self, idx: usize) -> f64 { | function signature/body; changes affect API behavior and control-flo
| 47 |             self.0[idx] | support logic; impact depends on neighboring block and data flow. |
| 48 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 |         fn set(&mut self, idx: usize, value: f64) { | function signature/body; changes affect API behavior and co
| 50 |             self.0[idx] = value; | support logic; impact depends on neighboring block and data flow. |
| 51 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 53 | (blank) | blank separator; no runtime behavior. |
| 54 |     struct UnitDrift; | data layout contract; field changes can break callers/serde/tests. |
| 55 |     impl DynamicsModel for UnitDrift { | implementation binding; behavior semantics and trait conformance live he
| 56 |         fn derivative(&self, x: &[f64], _t: f64) -> Vec<f64> { | function signature/body; changes affect API beha
| 57 |             vec![1.0; x.len()] | support logic; impact depends on neighboring block and data flow. |
| 58 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 59 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 60 | (blank) | blank separator; no runtime behavior. |
| 61 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 62 |     fn euler_integrator_advances_state() { | function signature/body; changes affect API behavior and control-flo
| 63 |         let model = UnitDrift; | support logic; impact depends on neighboring block and data flow. |
| 64 |         let integ = EulerIntegrator; | support logic; impact depends on neighboring block and data flow. |
| 65 |         let mut state = VecState(vec![0.0, 2.0, -1.0]); | support logic; impact depends on neighboring block and
| 66 |         integ.step(&model, &mut state, 0.0, 0.2); | support logic; impact depends on neighboring block and data
| 67 |         assert!((state.get(0) - 0.2).abs() < 1e-9); | support logic; impact depends on neighboring block and data
| 68 |         assert!((state.get(1) - 2.2).abs() < 1e-9); | support logic; impact depends on neighboring block and data
| 69 |         assert!((state.get(2) - -0.8).abs() < 1e-9); | support logic; impact depends on neighboring block and da
| 70 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/transmission.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 141 | Non-empty: 127 | Symbol count: 15

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| begin_shift | 3 |
| fmt | 1 |
| new | 1 |
| set_mode | 1 |
| update | 1 |
| gear_label | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 2 | enum | DriveMode | data/schema coupling and match/serde break risk |
| 11 | impl | std::fmt::Display for DriveMode | method behavior and trait semantics |
| 12 | fn | fmt | API and behavior coupling to callers/implementers |
| 26 | enum | ShiftEvent | data/schema coupling and match/serde break risk |
| 33 | struct | TransmissionEcu | data/schema coupling and match/serde break risk |
| 45 | const | UP_DRIVE | namespace and compile-time linkage |
| 46 | const | UP_SPORT | namespace and compile-time linkage |
| 47 | const | UP_ECO | namespace and compile-time linkage |
| 48 | const | DOWN_THR | namespace and compile-time linkage |
| 50 | impl | TransmissionEcu | method behavior and trait semantics |
| 51 | fn | new | API and behavior coupling to callers/implementers |
| 64 | fn | set_mode | API and behavior coupling to callers/implementers |
| 77 | fn | update | API and behavior coupling to callers/implementers |
| 122 | fn | begin_shift | API and behavior coupling to callers/implementers |
| 133 | fn | gear_label | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent tracea
| 2 | pub enum DriveMode { | state machine variants; changes ripple through matches and business logic. |
| 3 |     Park, | support logic; impact depends on neighboring block and data flow. |
| 4 |     Reverse, | support logic; impact depends on neighboring block and data flow. |
| 5 |     Neutral, | support logic; impact depends on neighboring block and data flow. |
| 6 |     Drive, | support logic; impact depends on neighboring block and data flow. |
| 7 |     Sport, | support logic; impact depends on neighboring block and data flow. |
| 8 |     Eco, | support logic; impact depends on neighboring block and data flow. |
| 9 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 10 | (blank) | blank separator; no runtime behavior. |
| 11 | impl std::fmt::Display for DriveMode { | implementation binding; behavior semantics and trait conformance live he
| 12 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 13 |         let s = match self { | support logic; impact depends on neighboring block and data flow. |
| 14 |             DriveMode::Park => "P", | support logic; impact depends on neighboring block and data flow. |
| 15 |             DriveMode::Reverse => "R", | support logic; impact depends on neighboring block and data flow. |
| 16 |             DriveMode::Neutral => "N", | support logic; impact depends on neighboring block and data flow. |
| 17 |             DriveMode::Drive => "D", | support logic; impact depends on neighboring block and data flow. |
| 18 |             DriveMode::Sport => "S", | support logic; impact depends on neighboring block and data flow. |
| 19 |             DriveMode::Eco => "E", | support logic; impact depends on neighboring block and data flow. |
| 20 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 21 |         write!(f, "{}", s) | protocol or I/O critical; changes may impact external integration correctness. |
| 22 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 26 | pub enum ShiftEvent { | state machine variants; changes ripple through matches and business logic. |
| 27 |     None, | support logic; impact depends on neighboring block and data flow. |
| 28 |     Up, | support logic; impact depends on neighboring block and data flow. |
| 29 |     Down, | support logic; impact depends on neighboring block and data flow. |
| 30 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 33 | pub struct TransmissionEcu { | data layout contract; field changes can break callers/serde/tests. |
| 34 |     pub gear: u8, // 1-8 | support logic; impact depends on neighboring block and data flow. |
| 35 |     pub mode: DriveMode, | support logic; impact depends on neighboring block and data flow. |
| 36 |     pub is_shifting: bool, | support logic; impact depends on neighboring block and data flow. |
| 37 |     pub shift_timer: f64, | support logic; impact depends on neighboring block and data flow. |
| 38 |     pub last_shift: ShiftEvent, | support logic; impact depends on neighboring block and data flow. |
| 39 |     pub tcu_temp: f64, | support logic; impact depends on neighboring block and data flow. |
| 40 |     pub torque_converter_slip: f64, // 0-1 | support logic; impact depends on neighboring block and data flow. |
| 41 |     pub oil_temp: f64, | support logic; impact depends on neighboring block and data flow. |
| 42 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 43 | (blank) | blank separator; no runtime behavior. |
| 44 | // [gear index 0..7] → upshift threshold RPM by mode | comment/documentation; changing affects understanding and
| 45 | const UP_DRIVE: [f64; 8] = [0.0, 3100.0, 3200.0, 3200.0, 3100.0, 3000.0, 3000.0, 3000.0]; | support logic; impact
| 46 | const UP_SPORT: [f64; 8] = [0.0, 5800.0, 5800.0, 5800.0, 5800.0, 5800.0, 5800.0, 5800.0]; | support logic; impact
| 47 | const UP_ECO: [f64; 8] = [0.0, 2100.0, 2200.0, 2200.0, 2000.0, 1900.0, 1900.0, 1900.0]; | support logic; impact
| 48 | const DOWN_THR: f64 = 1300.0; // downshift below this RPM | support logic; impact depends on neighboring block a
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 | impl TransmissionEcu { | implementation binding; behavior semantics and trait conformance live here. |
| 51 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 52 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 53 |             gear: 1, | support logic; impact depends on neighboring block and data flow. |
| 54 |             mode: DriveMode::Drive, | support logic; impact depends on neighboring block and data flow. |
| 55 |             is_shifting: false, | support logic; impact depends on neighboring block and data flow. |
| 56 |             shift_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 57 |             last_shift: ShiftEvent::None, | support logic; impact depends on neighboring block and data flow. |
| 58 |             tcu_temp: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 59 |             torque_converter_slip: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 60 |             oil_temp: 25.0, | support logic; impact depends on neighboring block and data flow. |
| 61 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 62 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 63 | (blank) | blank separator; no runtime behavior. |
| 64 |     pub fn set_mode(&mut self, mode: DriveMode) { | function signature/body; changes affect API behavior and con
| 65 |         self.mode = mode; | support logic; impact depends on neighboring block and data flow. |
| 66 |         match mode { | branch dispatcher; arm changes alter behavior class handling. |
| 67 |             DriveMode::Drive \| DriveMode::Eco \| DriveMode::Sport => { | support logic; impact depends on neigh
| 68 |                 if self.gear == 0 { | runtime gate; condition changes can enable/disable critical paths. |
| 69 |                     self.gear = 1; | support logic; impact depends on neighboring block and data flow. |
| 70 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 71 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 72 |             DriveMode::Park \| DriveMode::Neutral => {} | support logic; impact depends on neighboring block and
| 73 |             DriveMode::Reverse => self.gear = 0, | support logic; impact depends on neighboring block and data f
| 74 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

```
| 75 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 76 | (blank) | blank separator; no runtime behavior. |
| 77 |     pub fn update(&mut self, rpm: f64, throttle: f64, speed_kmh: f64, dt: f64) { | function signature/body; chang
| 78 |         // Shift cooldown | comment/documentation; changing affects understanding and intent traceability. |
| 79 |         if self.is_shifting { | runtime gate; condition changes can enable/disable critical paths. |
| 80 |             self.shift_timer -= dt; | support logic; impact depends on neighboring block and data flow. |
| 81 |             if self.shift_timer <= 0.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 82 |                 self.is_shifting = false; | support logic; impact depends on neighboring block and data flow. |
| 83 |                 self.last_shift = ShiftEvent::None; | support logic; impact depends on neighboring block and data
| 84 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 85 |             return; | support logic; impact depends on neighboring block and data flow. |
| 86 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 87 | (blank) | blank separator; no runtime behavior. |
| 88 |         if !matches!( | runtime gate; condition changes can enable/disable critical paths. |
| 89 |             self.mode, | support logic; impact depends on neighboring block and data flow. |
| 90 |             DriveMode::Drive \| DriveMode::Sport \| DriveMode::Eco | support logic; impact depends on neighboring
| 91 |         ) { | support logic; impact depends on neighboring block and data flow. |
| 92 |             return; | support logic; impact depends on neighboring block and data flow. |
| 93 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 94 |         if speed_kmh < 2.0 { | runtime gate; condition changes can enable/disable critical paths. |
| 95 |             return; | support logic; impact depends on neighboring block and data flow. |
| 96 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 97 | (blank) | blank separator; no runtime behavior. |
| 98 |         let gi = (self.gear as usize).saturating_sub(1).min(7); | support logic; impact depends on neighboring b
| 99 |         let upshift_rpm = match self.mode { | support logic; impact depends on neighboring block and data flow.
| 100 |             DriveMode::Sport => UP_SPORT[gi], | support logic; impact depends on neighboring block and data flo
| 101 |             DriveMode::Eco => UP_ECO[gi], | support logic; impact depends on neighboring block and data flow. |
| 102 |             _ => UP_DRIVE[gi], | support logic; impact depends on neighboring block and data flow. |
| 103 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 104 | (blank) | blank separator; no runtime behavior. |
| 105 |         if rpm > upshift_rpm && self.gear < 8 { | runtime gate; condition changes can enable/disable critical p
| 106 |             self.gear += 1; | support logic; impact depends on neighboring block and data flow. |
| 107 |             self.begin_shift(ShiftEvent::Up); | support logic; impact depends on neighboring block and data flo
| 108 |         } else if rpm < DOWN_THR && throttle > 0.1 && self.gear > 1 { | support logic; impact depends on neighb
| 109 |             self.gear -= 1; | support logic; impact depends on neighboring block and data flow. |
| 110 |             self.begin_shift(ShiftEvent::Down); | support logic; impact depends on neighboring block and data f
| 111 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 112 | (blank) | blank separator; no runtime behavior. |
| 113 |         // TC slip is highest at low speeds / high throttle | comment/documentation; changing affects understan
| 114 |         self.torque_converter_slip = | support logic; impact depends on neighboring block and data flow. |
| 115 |             ((1.0 - (speed_kmh / 40.0).min(1.0)) * throttle * 0.35).max(0.0); | support logic; impact depends o
| 116 | (blank) | blank separator; no runtime behavior. |
| 117 |         let tcu_target = 75.0 + throttle * 20.0; | support logic; impact depends on neighboring block and data
| 118 |         self.tcu_temp += (tcu_target - self.tcu_temp) * dt * 0.008; | support logic; impact depends on neighbor
| 119 |         self.oil_temp = self.tcu_temp - 2.0; | support logic; impact depends on neighboring block and data flow
| 120 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 121 | (blank) | blank separator; no runtime behavior. |
| 122 |     fn begin_shift(&mut self, event: ShiftEvent) { | function signature/body; changes affect API behavior and c
| 123 |         self.is_shifting = true; | support logic; impact depends on neighboring block and data flow. |
| 124 |         self.last_shift = event; | support logic; impact depends on neighboring block and data flow. |
| 125 |         // Sport shifts faster (~150ms), Drive ~220ms, Eco ~280ms | comment/documentation; changing affects und
| 126 |         self.shift_timer = match self.mode { | support logic; impact depends on neighboring block and data flow
| 127 |             DriveMode::Sport => 0.15, | support logic; impact depends on neighboring block and data flow. |
| 128 |             DriveMode::Eco => 0.28, | support logic; impact depends on neighboring block and data flow. |
| 129 |             _ => 0.22, | support logic; impact depends on neighboring block and data flow. |
| 130 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 131 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 132 | (blank) | blank separator; no runtime behavior. |
| 133 |     pub fn gear_label(&self) -> String { | function signature/body; changes affect API behavior and control-flo
| 134 |         match self.mode { | branch dispatcher; arm changes alter behavior class handling. |
| 135 |             DriveMode::Park => "P".into(), | support logic; impact depends on neighboring block and data flow.
| 136 |             DriveMode::Reverse => "R".into(), | support logic; impact depends on neighboring block and data flo
| 137 |             DriveMode::Neutral => "N".into(), | support logic; impact depends on neighboring block and data flo
| 138 |             _ => format!("{}{}", self.mode, self.gear), | support logic; impact depends on neighboring block an
| 139 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 140 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 141 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/uds.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 962 | Non-empty: 898 | Symbol count: 72

## Function Call Hotspots

| Function | Approx Calls In File |
|---|---:|
| nrc | 40 |
| push | 28 |
| len | 23 |
| log | 18 |
| new | 8 |
| process | 6 |
| dtc_to_3bytes | 6 |
| is_empty | 4 |
| svc_session_control | 2 |
| svc_ecu_reset | 2 |
| svc_clear_dtc | 2 |
| svc_read_dtc | 2 |
| svc_read_data_by_id | 2 |
| svc_security_access | 2 |
| svc_write_data_by_id | 2 |
| svc_routine_control | 2 |
| svc_request_download | 2 |
| svc_transfer_data | 2 |
| svc_request_transfer_exit | 2 |
| svc_tester_present | 2 |
| fmt | 1 |
| tick | 1 |
| snapshot | 1 |
| record_dtc_set | 1 |
| service_name | 1 |

## Symbol Index

| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 27 | const | MAX_DOWNLOAD_LEN | namespace and compile-time linkage |
| 32 | enum | DiagSession | data/schema coupling and match/serde break risk |
| 41 | impl | std::fmt::Display for DiagSession | method behavior and trait semantics |
| 42 | fn | fmt | API and behavior coupling to callers/implementers |
| 54 | enum | SecurityLevel | data/schema coupling and match/serde break risk |
| 69 | enum | Nrc | data/schema coupling and match/serde break risk |
| 93 | module | did | namespace and compile-time linkage |
| 95 | const | VIN | namespace and compile-time linkage |
| 96 | const | ECU_SERIAL | namespace and compile-time linkage |
| 97 | const | SW_VERSION | namespace and compile-time linkage |
| 98 | const | HW_VERSION | namespace and compile-time linkage |
| 99 | const | MANUF_DATE | namespace and compile-time linkage |
| 100 | const | PROGRAM_DATE | namespace and compile-time linkage |
| 101 | const | CALIBRATION_ID | namespace and compile-time linkage |
| 102 | const | FINGERPRINT | namespace and compile-time linkage |
| 105 | const | ENGINE_RPM | namespace and compile-time linkage |
| 106 | const | ENGINE_LOAD | namespace and compile-time linkage |
| 107 | const | COOLANT_TEMP | namespace and compile-time linkage |
| 108 | const | OIL_PRESSURE | namespace and compile-time linkage |
| 109 | const | BOOST_PRESSURE | namespace and compile-time linkage |
| 110 | const | FUEL_RATE | namespace and compile-time linkage |
| 111 | const | ENGINE_TORQUE | namespace and compile-time linkage |
| 112 | const | DPF_SOOT | namespace and compile-time linkage |
| 113 | const | DEF_LEVEL | namespace and compile-time linkage |
| 114 | const | SCR_EFFICIENCY | namespace and compile-time linkage |
| 115 | const | EXHAUST_TEMP | namespace and compile-time linkage |
| 116 | const | RAIL_PRESSURE | namespace and compile-time linkage |
| 117 | const | ENGINE_HOURS | namespace and compile-time linkage |
| 118 | const | BATTERY_VOLTAGE | namespace and compile-time linkage |
| 119 | const | VEHICLE_SPEED | namespace and compile-time linkage |
| 122 | const | IDLE_RPM_CAL | namespace and compile-time linkage |
| 123 | const | RATED_RPM_CAL | namespace and compile-time linkage |
| 124 | const | MAX_TORQUE_CAL | namespace and compile-time linkage |
| 125 | const | GOVERNOR_MODE_CAL | namespace and compile-time linkage |
| 126 | const | DPF_REGEN_THRESHOLD | namespace and compile-time linkage |
| 127 | const | DEF_WARNING_LEVEL | namespace and compile-time linkage |
| 128 | const | FUEL_MAP_SELECT | namespace and compile-time linkage |
| 129 | const | PTO_MAX_RPM | namespace and compile-time linkage |
| 130 | const | SERVICE_INTERVAL_H | namespace and compile-time linkage |
| 133 | const | ROUTINE_DPF_REGEN | namespace and compile-time linkage |
| 134 | const | ROUTINE_INJECTOR_TEST | namespace and compile-time linkage |
| 135 | const | ROUTINE_ACTUATOR_TEST | namespace and compile-time linkage |

```
| 136 | const | ROUTINE_RESET_ADAP | namespace and compile-time linkage |
| 142 | struct | FreezeFrame | data/schema coupling and match/serde break risk |
| 161 | enum | DownloadState | data/schema coupling and match/serde break risk |
| 170 | struct | UdsServer | data/schema coupling and match/serde break risk |
| 220 | struct | UdsEvent | data/schema coupling and match/serde break risk |
| 230 | enum | UdsEventResult | data/schema coupling and match/serde break risk |
| 236 | impl | UdsServer | method behavior and trait semantics |
| 237 | fn | new | API and behavior coupling to callers/implementers |
| 276 | fn | tick | API and behavior coupling to callers/implementers |
| 293 | fn | process | API and behavior coupling to callers/implementers |
| 320 | fn | svc_session_control | API and behavior coupling to callers/implementers |
| 353 | fn | svc_ecu_reset | API and behavior coupling to callers/implementers |
| 395 | fn | svc_clear_dtc | API and behavior coupling to callers/implementers |
| 428 | fn | svc_read_dtc | API and behavior coupling to callers/implementers |
| 504 | fn | svc_read_data_by_id | API and behavior coupling to callers/implementers |
| 554 | fn | svc_security_access | API and behavior coupling to callers/implementers |
| 640 | fn | svc_write_data_by_id | API and behavior coupling to callers/implementers |
| 684 | fn | svc_routine_control | API and behavior coupling to callers/implementers |
| 733 | fn | svc_request_download | API and behavior coupling to callers/implementers |
| 767 | fn | svc_transfer_data | API and behavior coupling to callers/implementers |
| 809 | fn | svc_request_transfer_exit | API and behavior coupling to callers/implementers |
| 830 | fn | svc_tester_present | API and behavior coupling to callers/implementers |
| 850 | fn | record_dtc_set | API and behavior coupling to callers/implementers |
| 879 | fn | nrc | API and behavior coupling to callers/implementers |
| 883 | fn | log | API and behavior coupling to callers/implementers |
| 905 | fn | service_name | API and behavior coupling to callers/implementers |
| 926 | fn | dtc_to_3bytes | API and behavior coupling to callers/implementers |
| 935 | module | tests | namespace and compile-time linkage |
| 939 | fn | session_default_relocks_security | API and behavior coupling to callers/implementers |
| 949 | fn | transfer_exit_rejects_incomplete_download | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! UDS — ISO 14229-1 Unified Diagnostic Services | comment/documentation; changing affects understanding and inte
| 2 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 3 | //! UDS is the standard diagnostic protocol used by ALL modern ECUs (automotive | comment/documentation; changing
| 4 | //! and off-highway). Every OEM tool (CAT ET, CNH EST, AGCO, Deere Service | comment/documentation; changing affe
| 5 | //! Advisor) sends UDS frames wrapped in J1939 or ISO-TP transport. | comment/documentation; changing affects unde
| 6 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 7 | //! Services implemented: | comment/documentation; changing affects understanding and intent traceability. |
| 8 | //!   0x10 DiagnosticSessionControl   — switch between default/extended/programming | comment/documentation; chang
| 9 | //!   0x11 ECUReset                    — hard reset, soft reset, key-off/on | comment/documentation; changing affe
| 10 | //!   0x14 ClearDiagnosticInformation  — erase DTCs | comment/documentation; changing affects understanding and i
| 11 | //!   0x19 ReadDTCInformation          — query active/stored/all DTCs | comment/documentation; changing affects u
| 12 | //!   0x22 ReadDataByIdentifier        — read live parameters by 2-byte DID | comment/documentation; changing af
| 13 | //!   0x27 SecurityAccess              — seed/key authentication for programming | comment/documentation; changin
| 14 | //!   0x2E WriteDataByIdentifier       — write calibration/config parameters | comment/documentation; changing a
| 15 | //!   0x31 RoutineControl              — start/stop routines (DPF regen, etc.) | comment/documentation; changing
| 16 | //!   0x3E TesterPresent               — keep session alive (sent every ≤5 s) | comment/documentation; changing a
| 17 | //!   0x34 RequestDownload             — begin firmware/cal download | comment/documentation; changing affects un
| 18 | //!   0x36 TransferData                — stream data blocks | comment/documentation; changing affects understandi
| 19 | //!   0x37 RequestTransferExit         — finish download | comment/documentation; changing affects understanding
| 20 | //! | comment/documentation; changing affects understanding and intent traceability. |
| 21 | //! Transport: J1939 TP (BAM or CMDT) or ISO-TP (ISO 15765-2) depending on bus. | comment/documentation; changing
| 22 | //! Max payload without transport: 7 bytes (with J1939 TP: up to 1785 bytes). | comment/documentation; changing a
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 25 | use rand::RngCore; | import wiring; changing path/name can break compilation and module linkage. |
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 | const MAX_DOWNLOAD_LEN: u32 = 16 * 1024 * 1024; | support logic; impact depends on neighboring block and data flo
| 28 | (blank) | blank separator; no runtime behavior. |
| 29 | // ■■ Diagnostic Session ■■■
```

```
| 42 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 43 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 44 |             DiagSession::Default => write!(f, "DEFAULT    "), | protocol or I/O critical; changes may impact exte
| 45 |             DiagSession::Extended => write!(f, "EXTENDED   "), | protocol or I/O critical; changes may impact ext
| 46 |             DiagSession::Programming => write!(f, "PROGRAMMING"), | protocol or I/O critical; changes may impact
| 47 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 48 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 50 | (blank) | blank separator; no runtime behavior. |
| 51 | // ■■ Security Level ■■■
```

```
| 118 |     pub const BATTERY_VOLTAGE: u16 = 0xDD0E; | support logic; impact depends on neighboring block and data flow
| 119 |     pub const VEHICLE_SPEED: u16 = 0xDD0F; | support logic; impact depends on neighboring block and data flow.
| 120 | (blank) | blank separator; no runtime behavior. |
| 121 |     // Calibration parameters (writeable in Extended session with security) | comment/documentation; changing af
| 122 |     pub const IDLE_RPM_CAL: u16 = 0xCA01; // Idle speed setpoint | support logic; impact depends on neighboring
| 123 |     pub const RATED_RPM_CAL: u16 = 0xCA02; // Rated speed limit | support logic; impact depends on neighboring b
| 124 |     pub const MAX_TORQUE_CAL: u16 = 0xCA03; // Max torque limit | support logic; impact depends on neighboring b
| 125 |     pub const GOVERNOR_MODE_CAL: u16 = 0xCA04; // Governor mode select | support logic; impact depends on neighb
| 126 |     pub const DPF_REGEN_THRESHOLD: u16 = 0xCA05; // DPF regen trigger % | support logic; impact depends on neigh
| 127 |     pub const DEF_WARNING_LEVEL: u16 = 0xCA06; // DEF warn threshold % | support logic; impact depends on neighb
| 128 |     pub const FUEL_MAP_SELECT: u16 = 0xCA07; // Fuel map (0=Std,1=Eco,2=Pwr) | support logic; impact depends on
| 129 |     pub const PTO_MAX_RPM: u16 = 0xCA08; // PTO speed limit | support logic; impact depends on neighboring block
| 130 |     pub const SERVICE_INTERVAL_H: u16 = 0xCA09; // Service interval hours | support logic; impact depends on nei
| 131 | (blank) | blank separator; no runtime behavior. |
| 132 |     // Routine IDs | comment/documentation; changing affects understanding and intent traceability. |
| 133 |     pub const ROUTINE_DPF_REGEN: u16 = 0xDF01; // Trigger active DPF regen | support logic; impact depends on ne
| 134 |     pub const ROUTINE_INJECTOR_TEST: u16 = 0xDF02; // Cylinder balance test | support logic; impact depends on n
| 135 |     pub const ROUTINE_ACTUATOR_TEST: u16 = 0xDF03; // VGT/EGR actuator test | support logic; impact depends on n
| 136 |     pub const ROUTINE_RESET_ADAP: u16 = 0xDF04; // Reset learned adaptations | support logic; impact depends on
| 137 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 138 | (blank) | blank separator; no runtime behavior. |
| 139 | // ■■ Freeze Frame — snapshot of all parameters when a DTC sets ■■■■■■■■■■■■■■■■■■ | comment/documentation; c
| 140 | (blank) | blank separator; no runtime behavior. |
| 141 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 142 | pub struct FreezeFrame { | data layout contract; field changes can break callers/serde/tests. |
| 143 |     pub dtc_spn: u32, | support logic; impact depends on neighboring block and data flow. |
| 144 |     pub dtc_fmi: u8, | support logic; impact depends on neighboring block and data flow. |
| 145 |     pub timestamp_h: f64, // engine hours when set | support logic; impact depends on neighboring block and data
| 146 |     pub engine_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 147 |     pub engine_load_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 148 |     pub vehicle_speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 149 |     pub coolant_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 150 |     pub oil_pressure_kpa: f64, | support logic; impact depends on neighboring block and data flow. |
| 151 |     pub fuel_rate_lph: f64, | support logic; impact depends on neighboring block and data flow. |
| 152 |     pub boost_kpa: f64, | support logic; impact depends on neighboring block and data flow. |
| 153 |     pub battery_v: f64, | support logic; impact depends on neighboring block and data flow. |
| 154 |     pub throttle_pct: f64, | support logic; impact depends on neighboring block and data flow. |
| 155 |     pub ambient_temp_c: f64, | support logic; impact depends on neighboring block and data flow. |
| 156 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 157 | (blank) | blank separator; no runtime behavior. |
| 158 | // ■■ Download state machine ■■■■■■■■■■■■■■■■■■
```

```
| 194 |     pub service_interval_h: f64, | support logic; impact depends on neighboring block and data flow. |
| 195 |     pub pto_max_rpm: f64, | support logic; impact depends on neighboring block and data flow. |
| 196 | (blank) | blank separator; no runtime behavior. |
| 197 |     // ■ Identification strings ■■■
```

```
| 270 |             event_log: VecDeque::new(), | support logic; impact depends on neighboring block and data flow. |
| 271 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 272 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 273 | (blank) | blank separator; no runtime behavior. |
| 274 |     // ━━━
```

```
| 346 |             format!("→ {:?}", self.session), | error propagation point; changes alter failure propagation seman
| 347 |         ); | support logic; impact depends on neighboring block and data flow. |
| 348 |         vec![0x50, sub, 0x00, 0x19, 0x01, 0xF4] // P2=25ms, P2*=500ms | support logic; impact depends on neighbo
| 349 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 350 | (blank) | blank separator; no runtime behavior. |
| 351 |     // ━━━
```

```
| 422 |         ); | support logic; impact depends on neighboring block and data flow. |
| 423 |         vec![0x54] | support logic; impact depends on neighboring block and data flow. |
| 424 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 425 | (blank) | blank separator; no runtime behavior. |
| 426 |     // ■■■
```

```
| 498 |         ); | support logic; impact depends on neighboring block and data flow. |
| 499 |         resp | support logic; impact depends on neighboring block and data flow. |
| 500 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 501 | (blank) | blank separator; no runtime behavior. |
| 502 |     // ━━━
```

```
| 574 |                     Some(sub), | support logic; impact depends on neighboring block and data flow. |
| 575 |                     None, | support logic; impact depends on neighboring block and data flow. |
| 576 |                     UdsEventResult::Positive, | protocol or I/O critical; changes may impact external integratio
| 577 |                     format!("Seed 0x{:08X}", self.security_seed), | support logic; impact depends on neighboring
| 578 |                 ); | support logic; impact depends on neighboring block and data flow. |
| 579 |                 resp | support logic; impact depends on neighboring block and data flow. |
| 580 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 581 |             // Even sub-functions = key send | comment/documentation; changing affects understanding and intent
| 582 |             sub if sub % 2 == 0 => { | support logic; impact depends on neighboring block and data flow. |
| 583 |                 if req.len() < 6 { | runtime gate; condition changes can enable/disable critical paths. |
| 584 |                     return self.nrc(0x27, Nrc::IncorrectMessageLength); | support logic; impact depends on neigh
| 585 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 586 |                 let key = ((req[2] as u32) << 24) | support logic; impact depends on neighboring block and data
| 587 |                     \| ((req[3] as u32) << 16) | support logic; impact depends on neighboring block and data flo
| 588 |                     \| ((req[4] as u32) << 8) | support logic; impact depends on neighboring block and data flow
| 589 |                     \| (req[5] as u32); | support logic; impact depends on neighboring block and data flow. |
| 590 |                 // Algorithm: XOR seed with constant (simplified — real uses AES/SHA) | comment/documentation; c
| 591 |                 let expected_key = self.security_seed ^ 0xDEADBEEF; | support logic; impact depends on neighbori
| 592 |                 if key == expected_key { | runtime gate; condition changes can enable/disable critical paths. |
| 593 |                     self.security = match sub { | support logic; impact depends on neighboring block and data fl
| 594 |                         0x02 => SecurityLevel::Level1, | support logic; impact depends on neighboring block and
| 595 |                         0x06 => SecurityLevel::Level3, | support logic; impact depends on neighboring block and
| 596 |                         0x12 => SecurityLevel::Level9, | support logic; impact depends on neighboring block and
| 597 |                         _ => SecurityLevel::Level1, | support logic; impact depends on neighboring block and dat
| 598 |                     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 599 |                     self.security_attempts = 0; | support logic; impact depends on neighboring block and data fl
| 600 |                     self.log( | support logic; impact depends on neighboring block and data flow. |
| 601 |                         ts, | support logic; impact depends on neighboring block and data flow. |
| 602 |                         0x27, | support logic; impact depends on neighboring block and data flow. |
| 603 |                         Some(sub), | support logic; impact depends on neighboring block and data flow. |
| 604 |                         None, | support logic; impact depends on neighboring block and data flow. |
| 605 |                         UdsEventResult::Positive, | protocol or I/O critical; changes may impact external integr
| 606 |                         format!("Unlocked {:?}", self.security), | error propagation point; changes alter failur
| 607 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 608 |                     vec![0x67, sub] | support logic; impact depends on neighboring block and data flow. |
| 609 |                 } else { | support logic; impact depends on neighboring block and data flow. |
| 610 |                     self.security_attempts += 1; | support logic; impact depends on neighboring block and data f
| 611 |                     if self.security_attempts >= 3 { | runtime gate; condition changes can enable/disable critic
| 612 |                         self.lockout_timer = 10.0; // 10-second lockout | support logic; impact depends on neigh
| 613 |                         self.log( | support logic; impact depends on neighboring block and data flow. |
| 614 |                             ts, | support logic; impact depends on neighboring block and data flow. |
| 615 |                             0x27, | support logic; impact depends on neighboring block and data flow. |
| 616 |                             Some(sub), | support logic; impact depends on neighboring block and data flow. |
| 617 |                             None, | support logic; impact depends on neighboring block and data flow. |
| 618 |                             UdsEventResult::SecurityFail, | protocol or I/O critical; changes may impact externa
| 619 |                             "LOCKED OUT after 3 failures".into(), | support logic; impact depends on neighboring
| 620 |                         ); | support logic; impact depends on neighboring block and data flow. |
| 621 |                         return self.nrc(0x27, Nrc::ExceededNumberOfAttempts); | support logic; impact depends on
| 622 |                     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 623 |                     self.log( | support logic; impact depends on neighboring block and data flow. |
| 624 |                         ts, | support logic; impact depends on neighboring block and data flow. |
| 625 |                         0x27, | support logic; impact depends on neighboring block and data flow. |
| 626 |                         Some(sub), | support logic; impact depends on neighboring block and data flow. |
| 627 |                         None, | support logic; impact depends on neighboring block and data flow. |
| 628 |                         UdsEventResult::SecurityFail, | protocol or I/O critical; changes may impact external in
| 629 |                         format!("Wrong key — attempt {}/3", self.security_attempts), | support logic; impact dep
| 630 |                     ); | support logic; impact depends on neighboring block and data flow. |
| 631 |                     self.nrc(0x27, Nrc::InvalidKey) | support logic; impact depends on neighboring block and dat
| 632 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 633 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 634 |             _ => self.nrc(0x27, Nrc::SubFunctionNotSupported), | support logic; impact depends on neighboring bl
| 635 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 636 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 637 | (blank) | blank separator; no runtime behavior. |
| 638 |     // ################################################################################ | comment/d
| 639 |     // Service 0x2E — Write Data By Identifier | comment/documentation; changing affects understanding and inter
| 640 |     fn svc_write_data_by_id(&mut self, req: &[u8], ts: f64) -> Vec<u8> { | function signature/body; changes affe
| 641 |         if req.len() < 3 { | runtime gate; condition changes can enable/disable critical paths. |
| 642 |             return self.nrc(0x2E, Nrc::IncorrectMessageLength); | support logic; impact depends on neighboring b
| 643 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 644 |         if self.session != DiagSession::Extended \|\| self.security < SecurityLevel::Level1 { | runtime gate; co
| 645 |             return self.nrc(0x2E, Nrc::SecurityAccessDenied); | support logic; impact depends on neighboring blo
| 646 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 647 |         let did = ((req[1] as u16) << 8) \| req[2] as u16; | support logic; impact depends on neighboring block
| 648 |         let data = &req[3..]; | support logic; impact depends on neighboring block and data flow. |
| 649 | (blank) | blank separator; no runtime behavior. |
```

```
| 650 |         match did { | branch dispatcher; arm changes alter behavior class handling. |
| 651 |             d if d == did::IDLE_RPM_CAL && data.len() >= 2 => { | support logic; impact depends on neighboring b
| 652 |                 self.idle_rpm_cal = (((data[0] as u16) << 8) \| data[1] as u16) as f64 * 0.125; | support logic
| 653 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 654 |             d if d == did::RATED_RPM_CAL && data.len() >= 2 => { | support logic; impact depends on neighboring
| 655 |                 self.rated_rpm_cal = (((data[0] as u16) << 8) \| data[1] as u16) as f64 * 0.125; | support logic
| 656 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 657 |             d if d == did::MAX_TORQUE_CAL && data.len() >= 2 => { | support logic; impact depends on neighboring
| 658 |                 self.max_torque_cal = (((data[0] as u16) << 8) \| data[1] as u16) as f64; | support logic; impac
| 659 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 660 |             d if d == did::DPF_REGEN_THRESHOLD && !data.is_empty() => { | support logic; impact depends on neigh
| 661 |                 self.dpf_regen_threshold = data[0] as f64 * 0.4; | support logic; impact depends on neighboring
| 662 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 663 |             d if d == did::DEF_WARNING_LEVEL && !data.is_empty() => { | support logic; impact depends on neighbo
| 664 |                 self.def_warning_level = data[0] as f64 * 0.4; | support logic; impact depends on neighboring b
| 665 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 666 |             d if d == did::FUEL_MAP_SELECT && !data.is_empty() => { | support logic; impact depends on neighbor
| 667 |                 self.fuel_map_select = data[0].min(2); | support logic; impact depends on neighboring block and
| 668 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 669 |             _ => return self.nrc(0x2E, Nrc::RequestOutOfRange), | support logic; impact depends on neighboring b
| 670 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 671 |         self.log( | support logic; impact depends on neighboring block and data flow. |
| 672 |             ts, | support logic; impact depends on neighboring block and data flow. |
| 673 |             0x2E, | support logic; impact depends on neighboring block and data flow. |
| 674 |             None, | support logic; impact depends on neighboring block and data flow. |
| 675 |             Some(did), | support logic; impact depends on neighboring block and data flow. |
| 676 |             UdsEventResult::Positive, | protocol or I/O critical; changes may impact external integration corre
| 677 |             format!("DID 0x{:04X} written", did), | support logic; impact depends on neighboring block and data
| 678 |         ); | support logic; impact depends on neighboring block and data flow. |
| 679 |         vec![0x6E, req[1], req[2]] | support logic; impact depends on neighboring block and data flow. |
| 680 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 681 | (blank) | blank separator; no runtime behavior. |
| 682 |     // ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ | comment/d
| 683 |     // Service 0x31 — Routine Control | comment/documentation; changing affects understanding and intent traceab
| 684 |     fn svc_routine_control(&mut self, req: &[u8], ts: f64) -> Vec<u8> { | function signature/body; changes affe
| 685 |         if req.len() < 4 { | runtime gate; condition changes can enable/disable critical paths. |
| 686 |             return self.nrc(0x31, Nrc::IncorrectMessageLength); | support logic; impact depends on neighboring b
| 687 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 688 |         if self.session == DiagSession::Default { | runtime gate; condition changes can enable/disable critical
| 689 |             return self.nrc(0x31, Nrc::ServiceNotSupportedInActiveSession); | support logic; impact depends on n
| 690 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 691 |         if self.security < SecurityLevel::Level1 { | runtime gate; condition changes can enable/disable critica
| 692 |             return self.nrc(0x31, Nrc::SecurityAccessDenied); | support logic; impact depends on neighboring blo
| 693 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 694 |         let sub = req[1]; // 0x01=start, 0x02=stop, 0x03=request result | support logic; impact depends on neigh
| 695 |         let rid_hi = req[2]; | support logic; impact depends on neighboring block and data flow. |
| 696 |         let rid_lo = req[3]; | support logic; impact depends on neighboring block and data flow. |
| 697 |         let rid: u16 = ((rid_hi as u16) << 8) \| rid_lo as u16; | support logic; impact depends on neighboring b
| 698 | (blank) | blank separator; no runtime behavior. |
| 699 |         match rid { | branch dispatcher; arm changes alter behavior class handling. |
| 700 |             r if r == did::ROUTINE_DPF_REGEN => match sub { | support logic; impact depends on neighboring bloc
| 701 |                 0x01 => { | support logic; impact depends on neighboring block and data flow. |
| 702 |                     self.dpf_regen_routine_active = true; | support logic; impact depends on neighboring block a
| 703 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 704 |                 0x02 => { | support logic; impact depends on neighboring block and data flow. |
| 705 |                     self.dpf_regen_routine_active = false; | support logic; impact depends on neighboring block
| 706 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 707 |                 _ => {} | support logic; impact depends on neighboring block and data flow. |
| 708 |             }, | support logic; impact depends on neighboring block and data flow. |
| 709 |             r if r == did::ROUTINE_INJECTOR_TEST => match sub { | support logic; impact depends on neighboring b
| 710 |                 0x01 => { | support logic; impact depends on neighboring block and data flow. |
| 711 |                     self.injector_test_active = true; | support logic; impact depends on neighboring block and c
| 712 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 713 |                 0x02 => { | support logic; impact depends on neighboring block and data flow. |
| 714 |                     self.injector_test_active = false; | support logic; impact depends on neighboring block and
| 715 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 716 |                 _ => {} | support logic; impact depends on neighboring block and data flow. |
| 717 |             }, | support logic; impact depends on neighboring block and data flow. |
| 718 |             _ => return self.nrc(0x31, Nrc::RequestOutOfRange), | support logic; impact depends on neighboring b
| 719 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 720 |         self.log( | support logic; impact depends on neighboring block and data flow. |
| 721 |             ts, | support logic; impact depends on neighboring block and data flow. |
| 722 |             0x31, | support logic; impact depends on neighboring block and data flow. |
| 723 |             Some(sub), | support logic; impact depends on neighboring block and data flow. |
| 724 |             Some(rid), | support logic; impact depends on neighboring block and data flow. |
| 725 |             UdsEventResult::Positive, | protocol or I/O critical; changes may impact external integration corre
```

```
| 726 |             format!("Routine 0x{:04X} sub={}", rid, sub), | support logic; impact depends on neighboring block a
| 727 |         ); | support logic; impact depends on neighboring block and data flow. |
| 728 |         vec![0x71, sub, rid_hi, rid_lo, 0x00] | support logic; impact depends on neighboring block and data flow
| 729 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 730 | (blank) | blank separator; no runtime behavior. |
| 731 |     // ■■■
```

```
| 802 |             UdsEventResult::Positive, | protocol or I/O critical; changes may impact external integration corre
| 803 |             format!("Block {} ({} bytes)", block_num, req.len() - 2), | support logic; impact depends on neighbo
| 804 |         ); | support logic; impact depends on neighboring block and data flow. |
| 805 |         vec![0x76, block_num] | support logic; impact depends on neighboring block and data flow. |
| 806 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 807 | (blank) | blank separator; no runtime behavior. |
| 808 |     // Service 0x37 — Request Transfer Exit | comment/documentation; changing affects understanding and intent
| 809 |     fn svc_request_transfer_exit(&mut self, _req: &[u8], ts: f64) -> Vec<u8> { | function signature/body; change
| 810 |         if self.download_state != DownloadState::Transferring { | runtime gate; condition changes can enable/dis
| 811 |             return self.nrc(0x37, Nrc::RequestSequenceError); | support logic; impact depends on neighboring bl
| 812 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 813 |         if self.download_received_len != self.download_expected_len { | runtime gate; condition changes can ena
| 814 |             self.download_state = DownloadState::Idle; | support logic; impact depends on neighboring block and
| 815 |             return self.nrc(0x37, Nrc::GeneralProgrammingFailure); | support logic; impact depends on neighbori
| 816 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 817 |         self.download_state = DownloadState::Complete; | support logic; impact depends on neighboring block and
| 818 |         self.log( | support logic; impact depends on neighboring block and data flow. |
| 819 |             ts, | support logic; impact depends on neighboring block and data flow. |
| 820 |             0x37, | support logic; impact depends on neighboring block and data flow. |
| 821 |             None, | support logic; impact depends on neighboring block and data flow. |
| 822 |             None, | support logic; impact depends on neighboring block and data flow. |
| 823 |             UdsEventResult::Positive, | protocol or I/O critical; changes may impact external integration corre
| 824 |             format!("{} bytes transferred", self.download_received_len), | support logic; impact depends on nei
| 825 |         ); | support logic; impact depends on neighboring block and data flow. |
| 826 |         vec![0x77] | support logic; impact depends on neighboring block and data flow. |
| 827 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 828 | (blank) | blank separator; no runtime behavior. |
| 829 |     // Service 0x3E — Tester Present (keep-alive) | comment/documentation; changing affects understanding and i
| 830 |     fn svc_tester_present(&mut self, req: &[u8], ts: f64) -> Vec<u8> { | function signature/body; changes affec
| 831 |         self.session_timer = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 832 |         let sub = if req.len() >= 2 { req[1] } else { 0x00 }; | support logic; impact depends on neighboring bl
| 833 |         // sub=0x80 = "zero sub function" (suppress positive response) | comment/documentation; changing affects
| 834 |         if sub == 0x80 { | runtime gate; condition changes can enable/disable critical paths. |
| 835 |             return Vec::new(); | support logic; impact depends on neighboring block and data flow. |
| 836 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 837 |         self.log( | support logic; impact depends on neighboring block and data flow. |
| 838 |             ts, | support logic; impact depends on neighboring block and data flow. |
| 839 |             0x3E, | support logic; impact depends on neighboring block and data flow. |
| 840 |             Some(sub), | support logic; impact depends on neighboring block and data flow. |
| 841 |             None, | support logic; impact depends on neighboring block and data flow. |
| 842 |             UdsEventResult::Positive, | protocol or I/O critical; changes may impact external integration corre
| 843 |             "Keep-alive".into(), | support logic; impact depends on neighboring block and data flow. |
| 844 |         ); | support logic; impact depends on neighboring block and data flow. |
| 845 |         vec![0x7E, sub] | support logic; impact depends on neighboring block and data flow. |
| 846 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 847 | (blank) | blank separator; no runtime behavior. |
| 848 |     // ■■■
```

```
| 878 |     // ■■■
```

```
| 954 | (blank) | blank separator; no runtime behavior. |
| 955 |         let td = s.process(&[0x36, 0x01, 0xAA, 0xBB, 0xCC], 0.1); | support logic; impact depends on neighboring
| 956 |         assert_eq!(td.first().copied(), Some(0x76)); | support logic; impact depends on neighboring block and da
| 957 | (blank) | blank separator; no runtime behavior. |
| 958 |         let exit = s.process(&[0x37, 0x00], 0.2); | support logic; impact depends on neighboring block and data
| 959 |         assert_eq!(exit.first().copied(), Some(0x7F)); | support logic; impact depends on neighboring block and
| 960 |         assert_eq!(exit.get(2).copied(), Some(Nrc::GeneralProgrammingFailure as u8)); | support logic; impact de
| 961 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 962 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/v2x_telematics.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 634 | Non-empty: 580 | Symbol count: 34

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 11 |
| push | 5 |
| push_event | 5 |
| fmt | 4 |
| is_empty | 3 |
| default | 2 |
| update | 2 |
| all | 2 |
| len | 1 |
| start_ota | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 13 | struct | BasicSafetyMessage | data/schema coupling and match/serde break risk |
| 32 | struct | BrakeStatus | data/schema coupling and match/serde break risk |
| 42 | struct | SignalPhaseAndTiming | data/schema coupling and match/serde break risk |
| 51 | enum | TrafficPhase | data/schema coupling and match/serde break risk |
| 58 | impl | std::fmt::Display for TrafficPhase | method behavior and trait semantics |
| 59 | fn | fmt | API and behavior coupling to callers/implementers |
| 72 | struct | WorkZoneAlert | data/schema coupling and match/serde break risk |
| 81 | struct | EmergencyAlert | data/schema coupling and match/serde break risk |
| 90 | enum | EmergencyType | data/schema coupling and match/serde break risk |
| 96 | struct | V2xModule | data/schema coupling and match/serde break risk |
| 128 | impl | Default for V2xModule | method behavior and trait semantics |
| 129 | fn | default | API and behavior coupling to callers/implementers |
| 134 | impl | V2xModule | method behavior and trait semantics |
| 135 | fn | new | API and behavior coupling to callers/implementers |
| 161 | fn | update | API and behavior coupling to callers/implementers |
| 291 | struct | TelematicsEvent | data/schema coupling and match/serde break risk |
| 300 | enum | TelEventType | data/schema coupling and match/serde break risk |
| 314 | enum | ConnectState | data/schema coupling and match/serde break risk |
| 321 | impl | std::fmt::Display for ConnectState | method behavior and trait semantics |
| 322 | fn | fmt | API and behavior coupling to callers/implementers |
| 332 | struct | TelematicsModule | data/schema coupling and match/serde break risk |
| 375 | struct | RemoteCommand | data/schema coupling and match/serde break risk |
| 384 | enum | RemoteCmdType | data/schema coupling and match/serde break risk |
| 394 | impl | std::fmt::Display for RemoteCmdType | method behavior and trait semantics |
| 395 | fn | fmt | API and behavior coupling to callers/implementers |
| 409 | struct | Geofence | data/schema coupling and match/serde break risk |
| 419 | enum | GeofenceAlert | data/schema coupling and match/serde break risk |
| 425 | impl | Default for TelematicsModule | method behavior and trait semantics |
| 426 | fn | default | API and behavior coupling to callers/implementers |
| 431 | impl | TelematicsModule | method behavior and trait semantics |
| 432 | fn | new | API and behavior coupling to callers/implementers |
| 482 | fn | update | API and behavior coupling to callers/implementers |
| 621 | fn | push_event | API and behavior coupling to callers/implementers |
| 628 | fn | start_ota | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
```

```
|---:|---|---|
| 1 | //! V2X — Vehicle-to-Everything communication (DSRC 5.9GHz + C-V2X LTE-V). | comment/documentation; changing affec
| 2 | //! Telematics — Fleet management, remote diagnostics, OTA update. | comment/documentation; changing affects under
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use std::collections::VecDeque; | import wiring; changing path/name can break compilation and module linkage. |
| 5 | use std::f64::consts::PI; | import wiring; changing path/name can break compilation and module linkage. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 | // ═══════════════════════════════════════════════════════════════════════════════ | comment/do
| 8 | // V2X Module | comment/documentation; changing affects understanding and intent traceability. |
| 9 | // ═══════════════════════════════════════════════════════════════════════════════ | comment/do
| 10 | (blank) | blank separator; no runtime behavior. |
| 11 | /// SAE J2735 Basic Safety Message | comment/documentation; changing affects understanding and intent traceabili
| 12 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 13 | pub struct BasicSafetyMessage { | data layout contract; field changes can break callers/serde/tests. |
| 14 |     pub msg_count: u8, | support logic; impact depends on neighboring block and data flow. |
| 15 |     pub vehicle_id: u32, | support logic; impact depends on neighboring block and data flow. |
| 16 |     pub lat_deg: f64, | support logic; impact depends on neighboring block and data flow. |
| 17 |     pub lon_deg: f64, | support logic; impact depends on neighboring block and data flow. |
| 18 |     pub elevation_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 19 |     pub speed_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 20 |     pub heading_deg: f64, | support logic; impact depends on neighboring block and data flow. |
| 21 |     pub accel_long: f64, // m/s² | support logic; impact depends on neighboring block and data flow. |
| 22 |     pub accel_lat: f64, | support logic; impact depends on neighboring block and data flow. |
| 23 |     pub accel_vert: f64, | support logic; impact depends on neighboring block and data flow. |
| 24 |     pub yaw_rate: f64, | support logic; impact depends on neighboring block and data flow. |
| 25 |     pub brake_status: BrakeStatus, | support logic; impact depends on neighboring block and data flow. |
| 26 |     pub size_length_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 27 |     pub size_width_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 28 |     pub timestamp_ms: u64, | support logic; impact depends on neighboring block and data flow. |
| 29 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 30 | (blank) | blank separator; no runtime behavior. |
| 31 | #[derive(Debug, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceability. |
| 32 | pub struct BrakeStatus { | data layout contract; field changes can break callers/serde/tests. |
| 33 |     pub brake_applied: bool, | support logic; impact depends on neighboring block and data flow. |
| 34 |     pub traction_control: bool, | support logic; impact depends on neighboring block and data flow. |
| 35 |     pub abs_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 36 |     pub stability: bool, | support logic; impact depends on neighboring block and data flow. |
| 37 |     pub aux_brakes: bool, | support logic; impact depends on neighboring block and data flow. |
| 38 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 39 | (blank) | blank separator; no runtime behavior. |
| 40 | /// SAE J2735 Signal Phase and Timing (SPaT) from infrastructure | comment/documentation; changing affects unders
| 41 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 42 | pub struct SignalPhaseAndTiming { | data layout contract; field changes can break callers/serde/tests. |
| 43 |     pub intersection_id: u32, | support logic; impact depends on neighboring block and data flow. |
| 44 |     pub movement_id: u8, | support logic; impact depends on neighboring block and data flow. |
| 45 |     pub phase_state: TrafficPhase, | support logic; impact depends on neighboring block and data flow. |
| 46 |     pub time_to_change_s: f64, | support logic; impact depends on neighboring block and data flow. |
| 47 |     pub distance_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 48 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trace
| 51 | pub enum TrafficPhase { | state machine variants; changes ripple through matches and business logic. |
| 52 |     Green, | support logic; impact depends on neighboring block and data flow. |
| 53 |     Yellow, | support logic; impact depends on neighboring block and data flow. |
| 54 |     Red, | support logic; impact depends on neighboring block and data flow. |
| 55 |     Unknown, | support logic; impact depends on neighboring block and data flow. |
| 56 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 57 | (blank) | blank separator; no runtime behavior. |
| 58 | impl std::fmt::Display for TrafficPhase { | implementation binding; behavior semantics and trait conformance live
| 59 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affec
| 60 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 61 |             TrafficPhase::Green => "GREEN", | support logic; impact depends on neighboring block and data flow.
| 62 |             TrafficPhase::Yellow => "YELLOW", | support logic; impact depends on neighboring block and data flow
| 63 |             TrafficPhase::Red => "RED  ", | support logic; impact depends on neighboring block and data flow. |
| 64 |             TrafficPhase::Unknown => "UNKNWN", | support logic; impact depends on neighboring block and data flo
| 65 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 66 |         .fmt(f) | support logic; impact depends on neighboring block and data flow. |
| 67 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 68 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 69 | (blank) | blank separator; no runtime behavior. |
| 70 | /// Work Zone Alert (roadwork ahead) | comment/documentation; changing affects understanding and intent traceabi
| 71 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 72 | pub struct WorkZoneAlert { | data layout contract; field changes can break callers/serde/tests. |
| 73 |     pub zone_id: u32, | support logic; impact depends on neighboring block and data flow. |
| 74 |     pub distance_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 75 |     pub speed_limit_kmh: f64, | support logic; impact depends on neighboring block and data flow. |
```

```
| 76 |     pub description: String, | support logic; impact depends on neighboring block and data flow. |
| 77 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 78 | (blank) | blank separator; no runtime behavior. |
| 79 | /// Emergency Vehicle Alert | comment/documentation; changing affects understanding and intent traceability. |
| 80 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 81 | pub struct EmergencyAlert { | data layout contract; field changes can break callers/serde/tests. |
| 82 |     pub vehicle_id: u32, | support logic; impact depends on neighboring block and data flow. |
| 83 |     pub vehicle_type: EmergencyType, | support logic; impact depends on neighboring block and data flow. |
| 84 |     pub distance_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 85 |     pub bearing_deg: f64, | support logic; impact depends on neighboring block and data flow. |
| 86 |     pub approaching: bool, | support logic; impact depends on neighboring block and data flow. |
| 87 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 88 | (blank) | blank separator; no runtime behavior. |
| 89 | #[derive(Debug, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceability. |
| 90 | pub enum EmergencyType { | state machine variants; changes ripple through matches and business logic. |
| 91 |     Ambulance, | support logic; impact depends on neighboring block and data flow. |
| 92 |     FireTruck, | support logic; impact depends on neighboring block and data flow. |
| 93 |     Police, | support logic; impact depends on neighboring block and data flow. |
| 94 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 95 | (blank) | blank separator; no runtime behavior. |
| 96 | pub struct V2xModule { | data layout contract; field changes can break callers/serde/tests. |
| 97 |     // ■ Own BSM ■■■
```

```
| 152 |             road_hazard_alert: false, | support logic; impact depends on neighboring block and data flow. |
| 153 |             bsm_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 154 |             sim_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 155 |             msg_count: 0, | support logic; impact depends on neighboring block and data flow. |
| 156 |             noise_t: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 157 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 158 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 159 | (blank) | blank separator; no runtime behavior. |
| 160 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent trac
| 161 |     pub fn update( | function signature/body; changes affect API behavior and control-flow. |
| 162 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 163 |         lat: f64, | support logic; impact depends on neighboring block and data flow. |
| 164 |         lon: f64, | support logic; impact depends on neighboring block and data flow. |
| 165 |         speed_ms: f64, | support logic; impact depends on neighboring block and data flow. |
| 166 |         heading: f64, | support logic; impact depends on neighboring block and data flow. |
| 167 |         _accel_lon: f64, | support logic; impact depends on neighboring block and data flow. |
| 168 |         _accel_lat: f64, | support logic; impact depends on neighboring block and data flow. |
| 169 |         _abs_active: bool, | support logic; impact depends on neighboring block and data flow. |
| 170 |         elapsed: f64, | support logic; impact depends on neighboring block and data flow. |
| 171 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 172 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 173 |         self.noise_t += dt; | support logic; impact depends on neighboring block and data flow. |
| 174 |         self.bsm_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 175 |         self.sim_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 176 |         let n = \|s: f64\| ((s * 127.1 + 311.7).sin() * 43758.5 % 1.0 - 0.5) * 2.0; | support logic; impact depe
| 177 |         let nt = self.noise_t; | support logic; impact depends on neighboring block and data flow. |
| 178 | (blank) | blank separator; no runtime behavior. |
| 179 |         // ■ Transmit BSM at 10 Hz ■■■
```

```
| 228 |             (TrafficPhase::Red, 90.0 - phase_cycle) | support logic; impact depends on neighboring block and dat
| 229 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 230 |         self.spat_messages.push(SignalPhaseAndTiming { | support logic; impact depends on neighboring block and
| 231 |             intersection_id: 0x1234, | support logic; impact depends on neighboring block and data flow. |
| 232 |             movement_id: 1, | support logic; impact depends on neighboring block and data flow. |
| 233 |             phase_state: phase, | support logic; impact depends on neighboring block and data flow. |
| 234 |             time_to_change_s: ttc, | support logic; impact depends on neighboring block and data flow. |
| 235 |             distance_m: spat_dist, | support logic; impact depends on neighboring block and data flow. |
| 236 |         }); | support logic; impact depends on neighboring block and data flow. |
| 237 | (blank) | blank separator; no runtime behavior. |
| 238 |         // ■ Work zone simulation ■■■
```

```
| 304 |     SpeedAlert, | support logic; impact depends on neighboring block and data flow. |
| 305 |     EngineOn, | support logic; impact depends on neighboring block and data flow. |
| 306 |     EngineOff, | support logic; impact depends on neighboring block and data flow. |
| 307 |     MaintenanceAlert, | support logic; impact depends on neighboring block and data flow. |
| 308 |     CrashDetect, | support logic; impact depends on neighboring block and data flow. |
| 309 |     OtaUpdateAvail, | support logic; impact depends on neighboring block and data flow. |
| 310 |     OtaUpdateComplete, | support logic; impact depends on neighboring block and data flow. |
| 311 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 312 | (blank) | blank separator; no runtime behavior. |
| 313 | #[derive(Debug, Clone, Copy, PartialEq)] | comment/documentation; changing affects understanding and intent trac
| 314 | pub enum ConnectState { | state machine variants; changes ripple through matches and business logic. |
| 315 |     Offline, | support logic; impact depends on neighboring block and data flow. |
| 316 |     Connecting, | support logic; impact depends on neighboring block and data flow. |
| 317 |     Connected4G, | support logic; impact depends on neighboring block and data flow. |
| 318 |     Connected5G, | support logic; impact depends on neighboring block and data flow. |
| 319 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 320 | (blank) | blank separator; no runtime behavior. |
| 321 | impl std::fmt::Display for ConnectState { | implementation binding; behavior semantics and trait conformance liv
| 322 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 323 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 324 |             ConnectState::Offline => write!(f, "OFFLINE   "), | protocol or I/O critical; changes may impact ext
| 325 |             ConnectState::Connecting => write!(f, "CONNECTING"), | protocol or I/O critical; changes may impact
| 326 |             ConnectState::Connected4G => write!(f, "4G LTE    "), | protocol or I/O critical; changes may impact
| 327 |             ConnectState::Connected5G => write!(f, "5G NR     "), | protocol or I/O critical; changes may impact
| 328 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 329 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 330 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 331 | (blank) | blank separator; no runtime behavior. |
| 332 | pub struct TelematicsModule { | data layout contract; field changes can break callers/serde/tests. |
| 333 |     // ■ Connectivity ■■■
```

```
| 380 |     pub executed: bool, | support logic; impact depends on neighboring block and data flow. |
| 381 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 382 | (blank) | blank separator; no runtime behavior. |
| 383 | #[derive(Debug, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceability.
| 384 | pub enum RemoteCmdType { | state machine variants; changes ripple through matches and business logic. |
| 385 |     RequestDtcs, | support logic; impact depends on neighboring block and data flow. |
| 386 |     ClearDtcs, | support logic; impact depends on neighboring block and data flow. |
| 387 |     UpdateFirmware, | support logic; impact depends on neighboring block and data flow. |
| 388 |     SetSpeedLimit, | support logic; impact depends on neighboring block and data flow. |
| 389 |     Immobilise, | support logic; impact depends on neighboring block and data flow. |
| 390 |     Unlock, | support logic; impact depends on neighboring block and data flow. |
| 391 |     RequestLocation, | support logic; impact depends on neighboring block and data flow. |
| 392 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 393 | (blank) | blank separator; no runtime behavior. |
| 394 | impl std::fmt::Display for RemoteCmdType { | implementation binding; behavior semantics and trait conformance li
| 395 |     fn fmt(&self, f: &mut std::fmt::Formatter<'_>) -> std::fmt::Result { | function signature/body; changes affe
| 396 |         match self { | branch dispatcher; arm changes alter behavior class handling. |
| 397 |             RemoteCmdType::RequestDtcs => write!(f, "REQUEST_DTC  "), | protocol or I/O critical; changes may in
| 398 |             RemoteCmdType::ClearDtcs => write!(f, "CLEAR_DTC    "), | protocol or I/O critical; changes may impa
| 399 |             RemoteCmdType::UpdateFirmware => write!(f, "OTA_UPDATE   "), | protocol or I/O critical; changes may
| 400 |             RemoteCmdType::SetSpeedLimit => write!(f, "SET_SPD_LIM  "), | protocol or I/O critical; changes may
| 401 |             RemoteCmdType::Immobilise => write!(f, "IMMOBILISE   "), | protocol or I/O critical; changes may imp
| 402 |             RemoteCmdType::Unlock => write!(f, "UNLOCK       "), | protocol or I/O critical; changes may impact
| 403 |             RemoteCmdType::RequestLocation => write!(f, "REQ_LOCATION "), | protocol or I/O critical; changes ma
| 404 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 405 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 406 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 407 | (blank) | blank separator; no runtime behavior. |
| 408 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 409 | pub struct Geofence { | data layout contract; field changes can break callers/serde/tests. |
| 410 |     pub id: u32, | support logic; impact depends on neighboring block and data flow. |
| 411 |     pub name: String, | support logic; impact depends on neighboring block and data flow. |
| 412 |     pub center_lat: f64, | support logic; impact depends on neighboring block and data flow. |
| 413 |     pub center_lon: f64, | support logic; impact depends on neighboring block and data flow. |
| 414 |     pub radius_m: f64, | support logic; impact depends on neighboring block and data flow. |
| 415 |     pub alert_type: GeofenceAlert, | support logic; impact depends on neighboring block and data flow. |
| 416 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 417 | (blank) | blank separator; no runtime behavior. |
| 418 | #[derive(Debug, Clone, Copy)] | comment/documentation; changing affects understanding and intent traceability.
| 419 | pub enum GeofenceAlert { | state machine variants; changes ripple through matches and business logic. |
| 420 |     Entry, | support logic; impact depends on neighboring block and data flow. |
| 421 |     Exit, | support logic; impact depends on neighboring block and data flow. |
| 422 |     Both, | support logic; impact depends on neighboring block and data flow. |
| 423 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 424 | (blank) | blank separator; no runtime behavior. |
| 425 | impl Default for TelematicsModule { | implementation binding; behavior semantics and trait conformance live here
| 426 |     fn default() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 427 |         Self::new() | support logic; impact depends on neighboring block and data flow. |
| 428 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 429 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 430 | (blank) | blank separator; no runtime behavior. |
| 431 | impl TelematicsModule { | implementation binding; behavior semantics and trait conformance live here. |
| 432 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 433 |         let geofences = vec![ | support logic; impact depends on neighboring block and data flow. |
| 434 |             Geofence { | support logic; impact depends on neighboring block and data flow. |
| 435 |                 id: 1, | support logic; impact depends on neighboring block and data flow. |
| 436 |                 name: "HQ Depot".into(), | support logic; impact depends on neighboring block and data flow. |
| 437 |                 center_lat: -22.810, | support logic; impact depends on neighboring block and data flow. |
| 438 |                 center_lon: -47.062, | support logic; impact depends on neighboring block and data flow. |
| 439 |                 radius_m: 500.0, | support logic; impact depends on neighboring block and data flow. |
| 440 |                 alert_type: GeofenceAlert::Both, | support logic; impact depends on neighboring block and data f
| 441 |             }, | support logic; impact depends on neighboring block and data flow. |
| 442 |             Geofence { | support logic; impact depends on neighboring block and data flow. |
| 443 |                 id: 2, | support logic; impact depends on neighboring block and data flow. |
| 444 |                 name: "Field Zone A".into(), | support logic; impact depends on neighboring block and data flow
| 445 |                 center_lat: -22.850, | support logic; impact depends on neighboring block and data flow. |
| 446 |                 center_lon: -47.090, | support logic; impact depends on neighboring block and data flow. |
| 447 |                 radius_m: 1000.0, | support logic; impact depends on neighboring block and data flow. |
| 448 |                 alert_type: GeofenceAlert::Entry, | support logic; impact depends on neighboring block and data
| 449 |             }, | support logic; impact depends on neighboring block and data flow. |
| 450 |         ]; | support logic; impact depends on neighboring block and data flow. |
| 451 |         TelematicsModule { | support logic; impact depends on neighboring block and data flow. |
| 452 |             connect_state: ConnectState::Connected4G, | support logic; impact depends on neighboring block and
| 453 |             signal_bars: 4, | support logic; impact depends on neighboring block and data flow. |
| 454 |             rsrp_dbm: -85.0, | support logic; impact depends on neighboring block and data flow. |
| 455 |             rsrq_db: -9.0, | support logic; impact depends on neighboring block and data flow. |
```

```
| 456 |             ping_ms: 28.0, | support logic; impact depends on neighboring block and data flow. |
| 457 |             download_kbps: 15000.0, | support logic; impact depends on neighboring block and data flow. |
| 458 |             upload_kbps: 5000.0, | support logic; impact depends on neighboring block and data flow. |
| 459 |             data_used_mb: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 460 |             fleet_server: "fleet.example.com:8443".into(), | support logic; impact depends on neighboring block
| 461 |             vehicle_id: "CAT-320-001".into(), | support logic; impact depends on neighboring block and data flo
| 462 |             fleet_unit_id: 1001, | support logic; impact depends on neighboring block and data flow. |
| 463 |             last_sync_s: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 464 |             report_interval_s: 30.0, | support logic; impact depends on neighboring block and data flow. |
| 465 |             pending_commands: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 466 |             ota_available: false, | support logic; impact depends on neighboring block and data flow. |
| 467 |             ota_version: String::new(), | support logic; impact depends on neighboring block and data flow. |
| 468 |             ota_size_mb: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 469 |             ota_progress_pct: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 470 |             ota_downloading: false, | support logic; impact depends on neighboring block and data flow. |
| 471 |             geofences, | support logic; impact depends on neighboring block and data flow. |
| 472 |             inside_geofence: None, | support logic; impact depends on neighboring block and data flow. |
| 473 |             events: VecDeque::new(), | support logic; impact depends on neighboring block and data flow. |
| 474 |             total_events_sent: 0, | support logic; impact depends on neighboring block and data flow. |
| 475 |             report_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 476 |             connect_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 477 |             noise_t: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 478 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 479 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 480 | (blank) | blank separator; no runtime behavior. |
| 481 |     #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent tra
| 482 |     pub fn update( | function signature/body; changes affect API behavior and control-flow. |
| 483 |         &mut self, | support logic; impact depends on neighboring block and data flow. |
| 484 |         lat: f64, | support logic; impact depends on neighboring block and data flow. |
| 485 |         lon: f64, | support logic; impact depends on neighboring block and data flow. |
| 486 |         speed_kmh: f64, | support logic; impact depends on neighboring block and data flow. |
| 487 |         engine_hours: f64, | support logic; impact depends on neighboring block and data flow. |
| 488 |         active_dtcs: usize, | support logic; impact depends on neighboring block and data flow. |
| 489 |         elapsed: f64, | support logic; impact depends on neighboring block and data flow. |
| 490 |         dt: f64, | support logic; impact depends on neighboring block and data flow. |
| 491 |     ) { | support logic; impact depends on neighboring block and data flow. |
| 492 |         self.noise_t += dt; | support logic; impact depends on neighboring block and data flow. |
| 493 |         self.report_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 494 |         self.connect_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 495 |         let n = \|s: f64\| ((s * 127.1 + 311.7).sin() * 43758.5 % 1.0 - 0.5) * 2.0; | support logic; impact dep
| 496 |         let nt = self.noise_t; | support logic; impact depends on neighboring block and data flow. |
| 497 | (blank) | blank separator; no runtime behavior. |
| 498 |         // ■ Signal quality simulation ■■■
```

```
| 532 |                 ), | support logic; impact depends on neighboring block and data flow. |
| 533 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 534 |             self.push_event(ev); | support logic; impact depends on neighboring block and data flow. |
| 535 |             self.total_events_sent += 1; | support logic; impact depends on neighboring block and data flow. |
| 536 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 537 | (blank) | blank separator; no runtime behavior. |
| 538 |         // ■ DTC alert ■■■
```

```
| 608 |                 self.ota_available = false; | support logic; impact depends on neighboring block and data flow.
| 609 |                 let ev = TelematicsEvent { | support logic; impact depends on neighboring block and data flow.
| 610 |                     timestamp: elapsed, | support logic; impact depends on neighboring block and data flow. |
| 611 |                     sent: true, | support logic; impact depends on neighboring block and data flow. |
| 612 |                     ack: true, | support logic; impact depends on neighboring block and data flow. |
| 613 |                     event_type: TelEventType::OtaUpdateComplete, | support logic; impact depends on neighboring
| 614 |                     description: format!("OTA {} installed", self.ota_version), | support logic; impact depends
| 615 |                 }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 616 |                 self.push_event(ev); | support logic; impact depends on neighboring block and data flow. |
| 617 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 618 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 619 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 620 | (blank) | blank separator; no runtime behavior. |
| 621 |     fn push_event(&mut self, ev: TelematicsEvent) { | function signature/body; changes affect API behavior and c
| 622 |         self.events.push_front(ev); | support logic; impact depends on neighboring block and data flow. |
| 623 |         if self.events.len() > 100 { | runtime gate; condition changes can enable/disable critical paths. |
| 624 |             self.events.pop_back(); | support logic; impact depends on neighboring block and data flow. |
| 625 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 626 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 627 | (blank) | blank separator; no runtime behavior. |
| 628 |     pub fn start_ota(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 629 |         if self.ota_available { | runtime gate; condition changes can enable/disable critical paths. |
| 630 |             self.ota_downloading = true; | support logic; impact depends on neighboring block and data flow. |
| 631 |             self.ota_progress_pct = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 632 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 633 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 634 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: src/vehicle.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 148 | Non-empty: 134 | Symbol count: 15

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 7 |
| in_path | 2 |
| push | 2 |
| velocity_ms | 1 |
| reset | 1 |
| ttc | 1 |
| update | 1 |
| lead_vehicle | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 3 | struct | VehicleState | data/schema coupling and match/serde break risk |
| 17 | impl | VehicleState | method behavior and trait semantics |
| 18 | fn | new | API and behavior coupling to callers/implementers |
| 35 | fn | velocity_ms | API and behavior coupling to callers/implementers |
| 39 | fn | reset | API and behavior coupling to callers/implementers |
| 46 | struct | TrafficObject | data/schema coupling and match/serde break risk |
| 56 | impl | TrafficObject | method behavior and trait semantics |
| 57 | fn | new | API and behavior coupling to callers/implementers |
| 70 | fn | ttc | API and behavior coupling to callers/implementers |
| 78 | fn | in_path | API and behavior coupling to callers/implementers |
| 84 | struct | TrafficSimulator | data/schema coupling and match/serde break risk |
| 90 | impl | TrafficSimulator | method behavior and trait semantics |
| 91 | fn | new | API and behavior coupling to callers/implementers |
| 104 | fn | update | API and behavior coupling to callers/implementers |
| 145 | fn | lead_vehicle | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | /// Physical state of the ego vehicle | comment/documentation; changing affects understanding and intent traceabil
| 2 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
```

```
| 3 | pub struct VehicleState { | data layout contract; field changes can break callers/serde/tests. |
| 4 |     pub velocity: f64,          // km/h longitudinal | support logic; impact depends on neighboring block and data
| 5 |     pub acceleration: f64,      // m/s² | support logic; impact depends on neighboring block and data flow. |
| 6 |     pub steering_angle: f64,    // degrees (-45..45) | support logic; impact depends on neighboring block and data
| 7 |     pub heading: f64,           // degrees 0-360 | support logic; impact depends on neighboring block and data flo
| 8 |     pub yaw_rate: f64,          // deg/s | support logic; impact depends on neighboring block and data flow. |
| 9 |     pub position_x: f64,        // meters | support logic; impact depends on neighboring block and data flow. |
| 10 |     pub position_y: f64,        // meters | support logic; impact depends on neighboring block and data flow. |
| 11 |     pub lateral_g: f64,         // g | support logic; impact depends on neighboring block and data flow. |
| 12 |     pub longitudinal_g: f64,    // g | support logic; impact depends on neighboring block and data flow. |
| 13 |     pub wheel_speeds: [f64; 4], // FL,FR,RL,RR  km/h | support logic; impact depends on neighboring block and da
| 14 |     pub wheel_slip: [f64; 4],   // -1..1 | support logic; impact depends on neighboring block and data flow. |
| 15 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 | impl VehicleState { | implementation binding; behavior semantics and trait conformance live here. |
| 18 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 19 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 20 |             velocity: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 21 |             acceleration: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 22 |             steering_angle: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 23 |             heading: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 24 |             yaw_rate: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 25 |             position_x: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 26 |             position_y: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 27 |             lateral_g: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 28 |             longitudinal_g: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 29 |             wheel_speeds: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 30 |             wheel_slip: [0.0; 4], | support logic; impact depends on neighboring block and data flow. |
| 31 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 32 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 |     /// Returns speed in m/s | comment/documentation; changing affects understanding and intent traceability. |
| 35 |     pub fn velocity_ms(&self) -> f64 { | function signature/body; changes affect API behavior and control-flow.
| 36 |         self.velocity / 3.6 | support logic; impact depends on neighboring block and data flow. |
| 37 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | (blank) | blank separator; no runtime behavior. |
| 39 |     pub fn reset(&mut self) { | function signature/body; changes affect API behavior and control-flow. |
| 40 |         *self = Self::new(); | support logic; impact depends on neighboring block and data flow. |
| 41 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 42 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 43 | (blank) | blank separator; no runtime behavior. |
| 44 | /// A simulated traffic participant | comment/documentation; changing affects understanding and intent traceabil
| 45 | #[derive(Debug, Clone)] | comment/documentation; changing affects understanding and intent traceability. |
| 46 | pub struct TrafficObject { | data layout contract; field changes can break callers/serde/tests. |
| 47 |     pub id: u32, | support logic; impact depends on neighboring block and data flow. |
| 48 |     pub distance: f64,       // m ahead (positive) / behind (negative) | support logic; impact depends on neighbo
| 49 |     pub lateral_offset: f64, // m from ego lane center | support logic; impact depends on neighboring block and
| 50 |     pub speed: f64,          // km/h | support logic; impact depends on neighboring block and data flow. |
| 51 |     pub closing_speed: f64,  // km/h (positive = approaching ego) | support logic; impact depends on neighboring
| 52 |     pub width: f64,          // m | support logic; impact depends on neighboring block and data flow. |
| 53 |     pub is_vehicle: bool, | support logic; impact depends on neighboring block and data flow. |
| 54 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 | (blank) | blank separator; no runtime behavior. |
| 56 | impl TrafficObject { | implementation binding; behavior semantics and trait conformance live here. |
| 57 |     pub fn new(id: u32, distance: f64, lateral_offset: f64, speed: f64) -> Self { | function signature/body; char
| 58 |         Self { | support logic; impact depends on neighboring block and data flow. |
| 59 |             id, | support logic; impact depends on neighboring block and data flow. |
| 60 |             distance, | support logic; impact depends on neighboring block and data flow. |
| 61 |             lateral_offset, | support logic; impact depends on neighboring block and data flow. |
| 62 |             speed, | support logic; impact depends on neighboring block and data flow. |
| 63 |             closing_speed: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 64 |             width: 1.8, | support logic; impact depends on neighboring block and data flow. |
| 65 |             is_vehicle: true, | support logic; impact depends on neighboring block and data flow. |
| 66 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 67 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 68 | (blank) | blank separator; no runtime behavior. |
| 69 |     /// Time to collision in seconds (0 = already colliding) | comment/documentation; changing affects understan
| 70 |     pub fn ttc(&self) -> f64 { | function signature/body; changes affect API behavior and control-flow. |
| 71 |         if self.closing_speed <= 0.0 \|\| self.distance <= 0.0 { | runtime gate; condition changes can enable/dis
| 72 |             return f64::INFINITY; | support logic; impact depends on neighboring block and data flow. |
| 73 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 74 |         self.distance / (self.closing_speed / 3.6) | support logic; impact depends on neighboring block and data
| 75 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 76 | (blank) | blank separator; no runtime behavior. |
| 77 |     /// Is this object in the ego vehicle path? | comment/documentation; changing affects understanding and inter
| 78 |     pub fn in_path(&self) -> bool { | function signature/body; changes affect API behavior and control-flow. |
```

```
| 79 |         self.lateral_offset.abs() < 1.5 && self.distance > 0.0 | support logic; impact depends on neighboring blo
| 80 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 81 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 82 | (blank) | blank separator; no runtime behavior. |
| 83 | /// Simple traffic simulation: one lead vehicle + side threats | comment/documentation; changing affects understa
| 84 | pub struct TrafficSimulator { | data layout contract; field changes can break callers/serde/tests. |
| 85 |     pub objects: Vec<TrafficObject>, | support logic; impact depends on neighboring block and data flow. |
| 86 |     lead_behavior_timer: f64, | support logic; impact depends on neighboring block and data flow. |
| 87 |     lead_target_speed: f64, | support logic; impact depends on neighboring block and data flow. |
| 88 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 89 | (blank) | blank separator; no runtime behavior. |
| 90 | impl TrafficSimulator { | implementation binding; behavior semantics and trait conformance live here. |
| 91 |     pub fn new() -> Self { | function signature/body; changes affect API behavior and control-flow. |
| 92 |         let mut sim = Self { | support logic; impact depends on neighboring block and data flow. |
| 93 |             objects: Vec::new(), | support logic; impact depends on neighboring block and data flow. |
| 94 |             lead_behavior_timer: 0.0, | support logic; impact depends on neighboring block and data flow. |
| 95 |             lead_target_speed: 80.0, | support logic; impact depends on neighboring block and data flow. |
| 96 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 97 |         // Lead vehicle | comment/documentation; changing affects understanding and intent traceability. |
| 98 |         sim.objects.push(TrafficObject::new(1, 80.0, 0.0, 80.0)); | support logic; impact depends on neighboring
| 99 |         // Static obstacle on side | comment/documentation; changing affects understanding and intent traceabili
| 100 |         sim.objects.push(TrafficObject::new(2, 30.0, 3.2, 0.0)); | support logic; impact depends on neighboring
| 101 |         sim | support logic; impact depends on neighboring block and data flow. |
| 102 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 103 | (blank) | blank separator; no runtime behavior. |
| 104 |     pub fn update(&mut self, ego_speed: f64, dt: f64) { | function signature/body; changes affect API behavior a
| 105 |         self.lead_behavior_timer += dt; | support logic; impact depends on neighboring block and data flow. |
| 106 | (blank) | blank separator; no runtime behavior. |
| 107 |         // Lead vehicle behavior changes every 8 seconds | comment/documentation; changing affects understanding
| 108 |         if self.lead_behavior_timer > 8.0 { | runtime gate; condition changes can enable/disable critical paths
| 109 |             self.lead_behavior_timer = 0.0; | support logic; impact depends on neighboring block and data flow.
| 110 |             // Alternate: slow down / speed up | comment/documentation; changing affects understanding and inter
| 111 |             self.lead_target_speed = if self.lead_target_speed > 60.0 { | support logic; impact depends on neigh
| 112 |                 30.0 | support logic; impact depends on neighboring block and data flow. |
| 113 |             } else { | support logic; impact depends on neighboring block and data flow. |
| 114 |                 90.0 | support logic; impact depends on neighboring block and data flow. |
| 115 |             }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 116 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 117 | (blank) | blank separator; no runtime behavior. |
| 118 |         for obj in &mut self.objects { | iteration logic; may alter timing, throughput, and safety constraints.
| 119 |             if !obj.is_vehicle { | runtime gate; condition changes can enable/disable critical paths. |
| 120 |                 continue; | support logic; impact depends on neighboring block and data flow. |
| 121 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 122 |             // Lead vehicle (id=1) tracks target speed | comment/documentation; changing affects understanding a
| 123 |             if obj.id == 1 { | runtime gate; condition changes can enable/disable critical paths. |
| 124 |                 let speed_diff = self.lead_target_speed - obj.speed; | support logic; impact depends on neighbo
| 125 |                 obj.speed += speed_diff * dt * 0.5; | support logic; impact depends on neighboring block and da
| 126 |                 obj.speed = obj.speed.max(0.0); | support logic; impact depends on neighboring block and data f
| 127 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 128 |             // Relative distance changes by closing speed | comment/documentation; changing affects understandin
| 129 |             let relative_speed_ms = (ego_speed - obj.speed) / 3.6; | support logic; impact depends on neighborin
| 130 |             obj.distance -= relative_speed_ms * dt; | support logic; impact depends on neighboring block and da
| 131 |             obj.closing_speed = ego_speed - obj.speed; | support logic; impact depends on neighboring block and
| 132 | (blank) | blank separator; no runtime behavior. |
| 133 |             // Teleport lead vehicle if it goes too far | comment/documentation; changing affects understanding
| 134 |             if obj.id == 1 { | runtime gate; condition changes can enable/disable critical paths. |
| 135 |                 if obj.distance < -20.0 { | runtime gate; condition changes can enable/disable critical paths.
| 136 |                     obj.distance = 180.0; | support logic; impact depends on neighboring block and data flow. |
| 137 |                     obj.speed = 80.0; | support logic; impact depends on neighboring block and data flow. |
| 138 |                 } else if obj.distance > 250.0 { | support logic; impact depends on neighboring block and data f
| 139 |                     obj.distance = 250.0; | support logic; impact depends on neighboring block and data flow. |
| 140 |                 } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 141 |             } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 142 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 143 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 144 | (blank) | blank separator; no runtime behavior. |
| 145 |     pub fn lead_vehicle(&self) -> Option<&TrafficObject> { | function signature/body; changes affect API behavio
| 146 |         self.objects.iter().find(\|o\| o.id == 1 && o.in_path()) | support logic; impact depends on neighboring
| 147 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 148 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: src/widgets.rs

Role: core simulation, domain logic, and ECU/network behavior

Line count: 327 | Non-empty: 303 | Symbol count: 8

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| new | 36 |
| arc_gauge | 1 |
| bar_gauge | 1 |
| warning_lamp | 1 |
| digital_readout | 1 |
| direction_selector | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 16 | fn | arc_gauge | API and behavior coupling to callers/implementers |
| 54 | const | START | namespace and compile-time linkage |
| 55 | const | SWEEP | namespace and compile-time linkage |
| 56 | const | N | namespace and compile-time linkage |
| 166 | fn | bar_gauge | API and behavior coupling to callers/implementers |
| 203 | fn | warning_lamp | API and behavior coupling to callers/implementers |
| 262 | fn | digital_readout | API and behavior coupling to callers/implementers |
| 283 | fn | direction_selector | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | //! Custom egui widgets: arc gauges, bar gauges, warning lamps, LED indicators. | comment/documentation; changing
| 2 | #![allow(float_literal_f32_fallback)] | comment/documentation; changing affects understanding and intent traceabi
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | use eframe::egui::{ | import wiring; changing path/name can break compilation and module linkage. |
| 5 |     self, Align2, Color32, FontId, Pos2, Rect, Response, Sense, Shape, Stroke, Ui, Vec2, | support logic; impact
| 6 | }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 7 | use std::f32::consts::PI; | import wiring; changing path/name can break compilation and module linkage. |
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | // ■■ Arc Gauge (tachometer / speedometer style) ■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■■ | comment/documentation;
| 10 | (blank) | blank separator; no runtime behavior. |
| 11 | /// Draws a semicircular arc gauge. | comment/documentation; changing affects understanding and intent traceabil
| 12 | ///   • size       = diameter in pixels (suggest 140–200) | comment/documentation; changing affects understandin
| 13 | ///   • warn/crit  = thresholds that change color (0.0 = ignore) | comment/documentation; changing affects under
| 14 | ///   • low_warn   = optional low-side warning (e.g. oil pressure) | comment/documentation; changing affects und
| 15 | #[allow(clippy::too_many_arguments)] | comment/documentation; changing affects understanding and intent traceabi
| 16 | pub fn arc_gauge( | function signature/body; changes affect API behavior and control-flow. |
| 17 |     ui: &mut Ui, | support logic; impact depends on neighboring block and data flow. |
| 18 |     value: f64, | support logic; impact depends on neighboring block and data flow. |
| 19 |     min: f64, | support logic; impact depends on neighboring block and data flow. |
| 20 |     max: f64, | support logic; impact depends on neighboring block and data flow. |
| 21 |     label: &str, | support logic; impact depends on neighboring block and data flow. |
| 22 |     unit: &str, | support logic; impact depends on neighboring block and data flow. |
| 23 |     size: f32, | support logic; impact depends on neighboring block and data flow. |
| 24 |     warn: f64, | support logic; impact depends on neighboring block and data flow. |
| 25 |     crit: f64, | support logic; impact depends on neighboring block and data flow. |
| 26 |     low_warn: Option<f64>, | support logic; impact depends on neighboring block and data flow. |
| 27 | ) -> Response { | support logic; impact depends on neighboring block and data flow. |
| 28 |     let (rect, response) = ui.allocate_exact_size(Vec2::splat(size), Sense::hover()); | support logic; impact de
| 29 |     if !ui.is_rect_visible(rect) { | runtime gate; condition changes can enable/disable critical paths. |
| 30 |         return response; | support logic; impact depends on neighboring block and data flow. |
| 31 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 32 | (blank) | blank separator; no runtime behavior. |
| 33 |     let painter = ui.painter_at(rect); | support logic; impact depends on neighboring block and data flow. |
| 34 |     let center = rect.center(); | support logic; impact depends on neighboring block and data flow. |
| 35 |     let r = size * 0.35; | support logic; impact depends on neighboring block and data flow. |
| 36 |     let frac = ((value - min) / (max - min).max(0.001)).clamp(0.0, 1.0) as f32; | support logic; impact depends
| 37 | (blank) | blank separator; no runtime behavior. |
| 38 |     // Color logic | comment/documentation; changing affects understanding and intent traceability. |
| 39 |     let value_color = if value >= crit { | support logic; impact depends on neighboring block and data flow. |
| 40 |         Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 41 |     } else if value >= warn && warn > 0.0 { | support logic; impact depends on neighboring block and data flow.
| 42 |         Color32::YELLOW | support logic; impact depends on neighboring block and data flow. |
| 43 |     } else if let Some(lo) = low_warn { | support logic; impact depends on neighboring block and data flow. |
| 44 |         if value <= lo { | runtime gate; condition changes can enable/disable critical paths. |
```

```
| 45 |             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 46 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 47 |             Color32::from_rgb(30, 210, 90) | support logic; impact depends on neighboring block and data flow. |
| 48 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 |     } else { | support logic; impact depends on neighboring block and data flow. |
| 50 |         Color32::from_rgb(30, 210, 90) | support logic; impact depends on neighboring block and data flow. |
| 51 |     }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 52 | (blank) | blank separator; no runtime behavior. |
| 53 |     // Arc: 135° → 405° (270° clockwise sweep) | comment/documentation; changing affects understanding and inter
| 54 |     const START: f32 = PI * 0.75; | support logic; impact depends on neighboring block and data flow. |
| 55 |     const SWEEP: f32 = PI * 1.50; | support logic; impact depends on neighboring block and data flow. |
| 56 |     const N: usize = 80; | support logic; impact depends on neighboring block and data flow. |
| 57 | (blank) | blank separator; no runtime behavior. |
| 58 |     // Background arc | comment/documentation; changing affects understanding and intent traceability. |
| 59 |     for i in 0..N { | iteration logic; may alter timing, throughput, and safety constraints. |
| 60 |         let a0 = START + (i as f32 / N as f32) * SWEEP; | support logic; impact depends on neighboring block and
| 61 |         let a1 = START + ((i + 1) as f32 / N as f32) * SWEEP; | support logic; impact depends on neighboring bloc
| 62 |         let p0 = center + Vec2::new(a0.cos() * r, a0.sin() * r); | support logic; impact depends on neighboring b
| 63 |         let p1 = center + Vec2::new(a1.cos() * r, a1.sin() * r); | support logic; impact depends on neighboring b
| 64 |         painter.line_segment([p0, p1], Stroke::new(4.0, Color32::from_gray(42))); | support logic; impact depends
| 65 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 66 | (blank) | blank separator; no runtime behavior. |
| 67 |     // Value arc (gradient green → yellow → red) | comment/documentation; changing affects understanding and in
| 68 |     let filled = ((frac * N as f32).floor() as usize).min(N); | support logic; impact depends on neighboring bloc
| 69 |     for i in 0..filled { | iteration logic; may alter timing, throughput, and safety constraints. |
| 70 |         let t = i as f32 / N as f32; | support logic; impact depends on neighboring block and data flow. |
| 71 |         let seg = if t < 0.60 { | support logic; impact depends on neighboring block and data flow. |
| 72 |             Color32::from_rgb(30, 210, 90) | support logic; impact depends on neighboring block and data flow. |
| 73 |         } else if t < 0.82 { | support logic; impact depends on neighboring block and data flow. |
| 74 |             Color32::YELLOW | support logic; impact depends on neighboring block and data flow. |
| 75 |         } else { | support logic; impact depends on neighboring block and data flow. |
| 76 |             Color32::RED | support logic; impact depends on neighboring block and data flow. |
| 77 |         }; | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 78 |         let a0 = START + t * SWEEP; | support logic; impact depends on neighboring block and data flow. |
| 79 |         let a1 = START + ((i + 1) as f32 / N as f32) * SWEEP; | support logic; impact depends on neighboring bloc
| 80 |         let p0 = center + Vec2::new(a0.cos() * r, a0.sin() * r); | support logic; impact depends on neighboring b
| 81 |         let p1 = center + Vec2::new(a1.cos() * r, a1.sin() * r); | support logic; impact depends on neighboring b
| 82 |         painter.line_segment([p0, p1], Stroke::new(5.5, seg)); | support logic; impact depends on neighboring blo
| 83 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 84 | (blank) | blank separator; no runtime behavior. |
| 85 |     // Warning tick mark | comment/documentation; changing affects understanding and intent traceability. |
| 86 |     if warn > min && warn < max { | runtime gate; condition changes can enable/disable critical paths. |
| 87 |         let wf = ((warn - min) / (max - min)) as f32; | support logic; impact depends on neighboring block and da
| 88 |         let wa = START + wf * SWEEP; | support logic; impact depends on neighboring block and data flow. |
| 89 |         let ti: Pos2 = center + Vec2::new(wa.cos() * (r - 8.0), wa.sin() * (r - 8.0)); | support logic; impact de
| 90 |         let to: Pos2 = center + Vec2::new(wa.cos() * (r + 4.0), wa.sin() * (r + 4.0)); | support logic; impact de
| 91 |         painter.line_segment([ti, to], Stroke::new(2.0, Color32::YELLOW)); | support logic; impact depends on nei
| 92 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 93 |     if crit > min && crit < max { | runtime gate; condition changes can enable/disable critical paths. |
| 94 |         let cf = ((crit - min) / (max - min)) as f32; | support logic; impact depends on neighboring block and da
| 95 |         let ca = START + cf * SWEEP; | support logic; impact depends on neighboring block and data flow. |
| 96 |         let ti: Pos2 = center + Vec2::new(ca.cos() * (r - 8.0), ca.sin() * (r - 8.0)); | support logic; impact de
| 97 |         let to: Pos2 = center + Vec2::new(ca.cos() * (r + 4.0), ca.sin() * (r + 4.0)); | support logic; impact de
| 98 |         painter.line_segment([ti, to], Stroke::new(2.0, Color32::RED)); | support logic; impact depends on neighb
| 99 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 100 | (blank) | blank separator; no runtime behavior. |
| 101 |     // Needle | comment/documentation; changing affects understanding and intent traceability. |
| 102 |     let na = START + frac * SWEEP; | support logic; impact depends on neighboring block and data flow. |
| 103 |     let tip: Pos2 = center + Vec2::new(na.cos() * r * 0.80, na.sin() * r * 0.80); | support logic; impact depend
| 104 |     let perp_a = na + PI * 0.5; | support logic; impact depends on neighboring block and data flow. |
| 105 |     let bl: Pos2 = center + Vec2::new(perp_a.cos() * 4.0, perp_a.sin() * 4.0); | support logic; impact depends o
| 106 |     let br: Pos2 = center - Vec2::new(perp_a.cos() * 4.0, perp_a.sin() * 4.0); | support logic; impact depends o
| 107 |     painter.add(Shape::convex_polygon( | support logic; impact depends on neighboring block and data flow. |
| 108 |         vec![tip, bl, br], | support logic; impact depends on neighboring block and data flow. |
| 109 |         Color32::WHITE, | support logic; impact depends on neighboring block and data flow. |
| 110 |         Stroke::NONE, | support logic; impact depends on neighboring block and data flow. |
| 111 |     )); | support logic; impact depends on neighboring block and data flow. |
| 112 | (blank) | blank separator; no runtime behavior. |
| 113 |     // Center cap | comment/documentation; changing affects understanding and intent traceability. |
| 114 |     painter.circle_filled(center, 7.0, Color32::from_gray(68)); | support logic; impact depends on neighboring b
| 115 |     painter.circle_stroke(center, 7.0, Stroke::new(1.5, Color32::from_gray(120))); | support logic; impact depen
| 116 | (blank) | blank separator; no runtime behavior. |
| 117 |     // Value text | comment/documentation; changing affects understanding and intent traceability. |
| 118 |     painter.text( | support logic; impact depends on neighboring block and data flow. |
| 119 |         center + Vec2::new(0.0, r * 0.28), | support logic; impact depends on neighboring block and data flow.
| 120 |         Align2::CENTER_CENTER, | support logic; impact depends on neighboring block and data flow. |
```

```
| 121 |         format!("{:.0}", value), | support logic; impact depends on neighboring block and data flow. |
| 122 |         FontId::proportional(size * 0.165), | support logic; impact depends on neighboring block and data flow.
| 123 |         value_color, | support logic; impact depends on neighboring block and data flow. |
| 124 |     ); | support logic; impact depends on neighboring block and data flow. |
| 125 |     // Unit | comment/documentation; changing affects understanding and intent traceability. |
| 126 |     painter.text( | support logic; impact depends on neighboring block and data flow. |
| 127 |         center + Vec2::new(0.0, r * 0.56), | support logic; impact depends on neighboring block and data flow.
| 128 |         Align2::CENTER_CENTER, | support logic; impact depends on neighboring block and data flow. |
| 129 |         unit, | support logic; impact depends on neighboring block and data flow. |
| 130 |         FontId::proportional(size * 0.080), | support logic; impact depends on neighboring block and data flow.
| 131 |         Color32::from_gray(155), | support logic; impact depends on neighboring block and data flow. |
| 132 |     ); | support logic; impact depends on neighboring block and data flow. |
| 133 |     // Label | comment/documentation; changing affects understanding and intent traceability. |
| 134 |     painter.text( | support logic; impact depends on neighboring block and data flow. |
| 135 |         center + Vec2::new(0.0, r * 0.85), | support logic; impact depends on neighboring block and data flow.
| 136 |         Align2::CENTER_CENTER, | support logic; impact depends on neighboring block and data flow. |
| 137 |         label, | support logic; impact depends on neighboring block and data flow. |
| 138 |         FontId::proportional(size * 0.095), | support logic; impact depends on neighboring block and data flow.
| 139 |         Color32::from_gray(200), | support logic; impact depends on neighboring block and data flow. |
| 140 |     ); | support logic; impact depends on neighboring block and data flow. |
| 141 | (blank) | blank separator; no runtime behavior. |
| 142 |     // Scale min/max | comment/documentation; changing affects understanding and intent traceability. |
| 143 |     let min_p: Pos2 = center + Vec2::new(START.cos() * (r + 14.0), START.sin() * (r + 14.0)); | support logic; i
| 144 |     let max_a = START + SWEEP; | support logic; impact depends on neighboring block and data flow. |
| 145 |     let max_p: Pos2 = center + Vec2::new(max_a.cos() * (r + 14.0), max_a.sin() * (r + 14.0)); | support logic; i
| 146 |     painter.text( | support logic; impact depends on neighboring block and data flow. |
| 147 |         min_p, | support logic; impact depends on neighboring block and data flow. |
| 148 |         Align2::CENTER_CENTER, | support logic; impact depends on neighboring block and data flow. |
| 149 |         format!("{:.0}", min), | support logic; impact depends on neighboring block and data flow. |
| 150 |         FontId::proportional(size * 0.075), | support logic; impact depends on neighboring block and data flow.
| 151 |         Color32::from_gray(90), | support logic; impact depends on neighboring block and data flow. |
| 152 |     ); | support logic; impact depends on neighboring block and data flow. |
| 153 |     painter.text( | support logic; impact depends on neighboring block and data flow. |
| 154 |         max_p, | support logic; impact depends on neighboring block and data flow. |
| 155 |         Align2::CENTER_CENTER, | support logic; impact depends on neighboring block and data flow. |
| 156 |         format!("{:.0}", max), | support logic; impact depends on neighboring block and data flow. |
| 157 |         FontId::proportional(size * 0.075), | support logic; impact depends on neighboring block and data flow.
| 158 |         Color32::from_gray(90), | support logic; impact depends on neighboring block and data flow. |
| 159 |     ); | support logic; impact depends on neighboring block and data flow. |
| 160 | (blank) | blank separator; no runtime behavior. |
| 161 |     response | support logic; impact depends on neighboring block and data flow. |
| 162 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 163 | (blank) | blank separator; no runtime behavior. |
| 164 | // ■■ Bar Gauge ■■■
```

```
| 197 |     }); | support logic; impact depends on neighboring block and data flow. |
| 198 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 199 | (blank) | blank separator; no runtime behavior. |
| 200 | // ■■ Warning Lamp (LED style) ■■■
```

```
| 273 |                 .monospace() | support logic; impact depends on neighboring block and data flow. |
| 274 |                 .color(color) | support logic; impact depends on neighboring block and data flow. |
| 275 |                 .strong(), | support logic; impact depends on neighboring block and data flow. |
| 276 |         ); | support logic; impact depends on neighboring block and data flow. |
| 277 |     }); | support logic; impact depends on neighboring block and data flow. |
| 278 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 279 | (blank) | blank separator; no runtime behavior. |
| 280 | // ■■ Direction Selector ■■■
```

```
| mock_adapter_blocks_write_by_default_and_reads_injected_frame | 1 |
| default | 1 |
| inject_rx | 1 |
| read_frame | 1 |
| mock_adapter_allows_write_when_policy_satisfied | 1 |
| mock_adapter_dry_run_logs_without_physical_tx | 1 |
| write_intent_enabled | 1 |
| mock_adapter_rate_limit_blocks_burst | 1 |
| allowlist_parse_mask_accepts_hex | 1 |
| parse_mask | 1 |
| replay_can_record_keeps_explicit_timestamp | 1 |
| frame_to_record | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 4 | module | io | namespace and compile-time linkage |
| 10 | fn | write_test_allowlist | API and behavior coupling to callers/implementers |
| 25 | fn | hw_config_parses_cli_flags | API and behavior coupling to callers/implementers |
| 75 | fn | mock_adapter_blocks_write_by_default_and_reads_injected_frame | API and behavior coupling to callers/impleme
| 97 | fn | mock_adapter_allows_write_when_policy_satisfied | API and behavior coupling to callers/implementers |
| 125 | fn | mock_adapter_dry_run_logs_without_physical_tx | API and behavior coupling to callers/implementers |
| 157 | fn | mock_adapter_rate_limit_blocks_burst | API and behavior coupling to callers/implementers |
| 189 | fn | allowlist_parse_mask_accepts_hex | API and behavior coupling to callers/implementers |
| 195 | fn | replay_can_record_keeps_explicit_timestamp | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #![allow(dead_code)] | comment/documentation; changing affects understanding and intent traceability. |
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | #[path = "../src/io/mod.rs"] | comment/documentation; changing affects understanding and intent traceability. |
| 4 | mod io; | module declaration; changing can break namespace graph. |
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use io::hw::{CanFrame, Frame, HardwareInterface, HwConfig, HwError, HwMode}; | import wiring; changing path/name c
| 7 | use io::mock::MockAdapter; | import wiring; changing path/name can break compilation and module linkage. |
| 8 | use std::fs; | import wiring; changing path/name can break compilation and module linkage. |
| 9 | (blank) | blank separator; no runtime behavior. |
| 10 | fn write_test_allowlist(path: &std::path::Path) { | function signature/body; changes affect API behavior and cont
| 11 |     let content = r#"[ | support logic; impact depends on neighboring block and data flow. |
| 12 |     { | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 13 |         "type": "can", | support logic; impact depends on neighboring block and data flow. |
| 14 |         "id": 419385573, | support logic; impact depends on neighboring block and data flow. |
| 15 |         "mask": "0x1FFFFFFF", | support logic; impact depends on neighboring block and data flow. |
| 16 |         "allowed_bytes": [[0, 8]], | support logic; impact depends on neighboring block and data flow. |
| 17 |         "max_rate_per_sec": 10, | support logic; impact depends on neighboring block and data flow. |
| 18 |         "description": "allow test can id" | support logic; impact depends on neighboring block and data flow. |
| 19 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 20 | ]"#; | support logic; impact depends on neighboring block and data flow. |
| 21 |     fs::write(path, content).expect("allowlist write"); | panic boundary; edits alter crash behavior and resilien
| 22 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 25 | fn hw_config_parses_cli_flags() { | function signature/body; changes affect API behavior and control-flow. |
| 26 |     let args = vec![ | support logic; impact depends on neighboring block and data flow. |
| 27 |         "simubench".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 28 |         "--hw-mode=live".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 29 |         "--dry-run".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 30 |         "--allowlist=./allowlist.json".to_string(), | support logic; impact depends on neighboring block and data
| 31 |         "--vendor-name=cat_comm".to_string(), | support logic; impact depends on neighboring block and data flow
| 32 |         "--vendor-template-dir=./vendor_template".to_string(), | support logic; impact depends on neighboring bl
| 33 |         "--vendor-bridge-exe=./vendor_template/cat_comm_bridge.exe".to_string(), | support logic; impact depends
| 34 |         "--vendor-bridge-timeout-ms=2200".to_string(), | support logic; impact depends on neighboring block and
| 35 |         "--enable-write".to_string(), | protocol or I/O critical; changes may impact external integration correc
| 36 |         "--noninteractive-approved".to_string(), | support logic; impact depends on neighboring block and data f
| 37 |         "--uds-retries=5".to_string(), | protocol or I/O critical; changes may impact external integration corre
| 38 |         "--uds-timeout-p2-ms=1300".to_string(), | protocol or I/O critical; changes may impact external integrati
| 39 |         "--uds-timeout-p2star-ms=4500".to_string(), | protocol or I/O critical; changes may impact external integ
| 40 |         "--uds-st-min-ms=10".to_string(), | protocol or I/O critical; changes may impact external integration co
| 41 |         "--uds-block-size=8".to_string(), | protocol or I/O critical; changes may impact external integration co
| 42 |         "--j1939-idle-guard-ms=3000".to_string(), | support logic; impact depends on neighboring block and data
| 43 |         "--keep-channels-alive=false".to_string(), | support logic; impact depends on neighboring block and data
| 44 |     ]; | support logic; impact depends on neighboring block and data flow. |
```

```
| 45 | (blank) | blank separator; no runtime behavior. |
| 46 |     let cfg = HwConfig::from_cli_args(&args).expect("valid args"); | panic boundary; edits alter crash behavior a
| 47 |     assert_eq!(cfg.mode, HwMode::Live); | support logic; impact depends on neighboring block and data flow. |
| 48 |     assert!(cfg.dry_run); | support logic; impact depends on neighboring block and data flow. |
| 49 |     assert_eq!(cfg.vendor_name.as_deref(), Some("cat_comm")); | support logic; impact depends on neighboring bloc
| 50 |     assert_eq!( | support logic; impact depends on neighboring block and data flow. |
| 51 |         cfg.vendor_template_dir.as_deref(), | support logic; impact depends on neighboring block and data flow.
| 52 |         Some(std::path::Path::new("./vendor_template")) | support logic; impact depends on neighboring block and
| 53 |     ); | support logic; impact depends on neighboring block and data flow. |
| 54 |     assert_eq!( | support logic; impact depends on neighboring block and data flow. |
| 55 |         cfg.vendor_bridge_exe.as_deref(), | support logic; impact depends on neighboring block and data flow. |
| 56 |         Some(std::path::Path::new("./vendor_template/cat_comm_bridge.exe")) | support logic; impact depends on ne
| 57 |     ); | support logic; impact depends on neighboring block and data flow. |
| 58 |     assert_eq!(cfg.vendor_bridge_timeout_ms, 2200); | support logic; impact depends on neighboring block and data
| 59 |     assert_eq!( | support logic; impact depends on neighboring block and data flow. |
| 60 |         cfg.allowlist_path.as_deref(), | support logic; impact depends on neighboring block and data flow. |
| 61 |         Some(std::path::Path::new("./allowlist.json")) | support logic; impact depends on neighboring block and
| 62 |     ); | support logic; impact depends on neighboring block and data flow. |
| 63 |     // dry-run must keep writes disabled effectively | comment/documentation; changing affects understanding and
| 64 |     assert!(!cfg.write_effectively_enabled()); | protocol or I/O critical; changes may impact external integratio
| 65 |     assert_eq!(cfg.uds_retry_count, 5); | protocol or I/O critical; changes may impact external integration corre
| 66 |     assert_eq!(cfg.uds_timeout_p2_ms, 1300); | protocol or I/O critical; changes may impact external integration
| 67 |     assert_eq!(cfg.uds_timeout_p2star_ms, 4500); | protocol or I/O critical; changes may impact external integrat
| 68 |     assert_eq!(cfg.uds_st_min_ms, 10); | protocol or I/O critical; changes may impact external integration correc
| 69 |     assert_eq!(cfg.uds_block_size, 8); | protocol or I/O critical; changes may impact external integration correc
| 70 |     assert_eq!(cfg.j1939_idle_guard_ms, 3000); | support logic; impact depends on neighboring block and data flow
| 71 |     assert!(!cfg.keep_channels_alive); | support logic; impact depends on neighboring block and data flow. |
| 72 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 73 | (blank) | blank separator; no runtime behavior. |
| 74 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 75 | fn mock_adapter_blocks_write_by_default_and_reads_injected_frame() { | function signature/body; changes affect AP
| 76 |     let cfg = HwConfig::default(); | support logic; impact depends on neighboring block and data flow. |
| 77 |     let mut ad = MockAdapter::new(); | support logic; impact depends on neighboring block and data flow. |
| 78 |     ad.init(&cfg).expect("init ok"); | panic boundary; edits alter crash behavior and resilience. |
| 79 | (blank) | blank separator; no runtime behavior. |
| 80 |     let f = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 81 |         id: 0x18FF50E5, | support logic; impact depends on neighboring block and data flow. |
| 82 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 83 |         data: [1, 2, 3, 4, 5, 6, 7, 8], | support logic; impact depends on neighboring block and data flow. |
| 84 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 85 |         timestamp_ms: Some(1), | support logic; impact depends on neighboring block and data flow. |
| 86 |     }); | support logic; impact depends on neighboring block and data flow. |
| 87 | (blank) | blank separator; no runtime behavior. |
| 88 |     let err = ad.send_frame(f.clone()).expect_err("write must be blocked"); | protocol or I/O critical; changes m
| 89 |     assert!(matches!(err, HwError::WriteBlockedAllowlist)); | support logic; impact depends on neighboring block
| 90 | (blank) | blank separator; no runtime behavior. |
| 91 |     ad.inject_rx(f.clone()); | support logic; impact depends on neighboring block and data flow. |
| 92 |     let got = ad.read_frame().expect("read injected frame"); | panic boundary; edits alter crash behavior and res
| 93 |     assert_eq!(got, f); | support logic; impact depends on neighboring block and data flow. |
| 94 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 95 | (blank) | blank separator; no runtime behavior. |
| 96 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 97 | fn mock_adapter_allows_write_when_policy_satisfied() { | function signature/body; changes affect API behavior and
| 98 |     let temp = std::env::temp_dir().join("allowlist_io_mock.json"); | support logic; impact depends on neighborin
| 99 |     write_test_allowlist(&temp); | protocol or I/O critical; changes may impact external integration correctness
| 100 | (blank) | blank separator; no runtime behavior. |
| 101 |     let args = vec![ | support logic; impact depends on neighboring block and data flow. |
| 102 |         "simubench".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 103 |         "--enable-write".to_string(), | protocol or I/O critical; changes may impact external integration correc
| 104 |         format!("--allowlist={}", temp.display()), | support logic; impact depends on neighboring block and data
| 105 |         "--noninteractive-approved".to_string(), | support logic; impact depends on neighboring block and data f
| 106 |         "--dry-run=false".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 107 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 108 |     let cfg = HwConfig::from_cli_args(&args).expect("valid args"); | panic boundary; edits alter crash behavior
| 109 |     assert!(cfg.write_effectively_enabled()); | protocol or I/O critical; changes may impact external integratio
| 110 | (blank) | blank separator; no runtime behavior. |
| 111 |     let mut ad = MockAdapter::new(); | support logic; impact depends on neighboring block and data flow. |
| 112 |     ad.init(&cfg).expect("init ok"); | panic boundary; edits alter crash behavior and resilience. |
| 113 |     let f = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 114 |         id: 0x18FF50E5, | support logic; impact depends on neighboring block and data flow. |
| 115 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 116 |         data: [0; 8], | support logic; impact depends on neighboring block and data flow. |
| 117 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 118 |         timestamp_ms: None, | support logic; impact depends on neighboring block and data flow. |
| 119 |     }); | support logic; impact depends on neighboring block and data flow. |
| 120 |     ad.send_frame(f).expect("write allowed"); | panic boundary; edits alter crash behavior and resilience. |
```

```
| 121 |     assert_eq!(ad.tx_queue.len(), 1); | support logic; impact depends on neighboring block and data flow. |
| 122 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 123 | (blank) | blank separator; no runtime behavior. |
| 124 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 125 | fn mock_adapter_dry_run_logs_without_physical_tx() { | function signature/body; changes affect API behavior and
| 126 |     let temp_allow = std::env::temp_dir().join("allowlist_io_mock_dry_run.json"); | support logic; impact depen
| 127 |     let temp_log_dir = std::env::temp_dir().join("autobreaking_hw_logs_dry_run"); | support logic; impact depen
| 128 |     write_test_allowlist(&temp_allow); | protocol or I/O critical; changes may impact external integration corre
| 129 | (blank) | blank separator; no runtime behavior. |
| 130 |     let args = vec![ | support logic; impact depends on neighboring block and data flow. |
| 131 |         "simubench".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 132 |         "--enable-write".to_string(), | protocol or I/O critical; changes may impact external integration corre
| 133 |         format!("--allowlist={}", temp_allow.display()), | support logic; impact depends on neighboring block a
| 134 |         "--noninteractive-approved".to_string(), | support logic; impact depends on neighboring block and data
| 135 |         "--dry-run".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 136 |         format!("--log-dir={}", temp_log_dir.display()), | support logic; impact depends on neighboring block a
| 137 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 138 |     let cfg = HwConfig::from_cli_args(&args).expect("valid args"); | panic boundary; edits alter crash behavior
| 139 |     assert!(!cfg.write_effectively_enabled()); | protocol or I/O critical; changes may impact external integrati
| 140 |     assert!(cfg.write_intent_enabled()); | protocol or I/O critical; changes may impact external integration co
| 141 | (blank) | blank separator; no runtime behavior. |
| 142 |     let mut ad = MockAdapter::new(); | support logic; impact depends on neighboring block and data flow. |
| 143 |     ad.init(&cfg).expect("init ok"); | panic boundary; edits alter crash behavior and resilience. |
| 144 |     let f = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 145 |         id: 0x18FF50E5, | support logic; impact depends on neighboring block and data flow. |
| 146 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 147 |         data: [0; 8], | support logic; impact depends on neighboring block and data flow. |
| 148 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 149 |         timestamp_ms: None, | support logic; impact depends on neighboring block and data flow. |
| 150 |     }); | support logic; impact depends on neighboring block and data flow. |
| 151 | (blank) | blank separator; no runtime behavior. |
| 152 |     ad.send_frame(f).expect("dry-run should allow intent"); | panic boundary; edits alter crash behavior and res
| 153 |     assert_eq!(ad.tx_queue.len(), 0); | support logic; impact depends on neighboring block and data flow. |
| 154 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 155 | (blank) | blank separator; no runtime behavior. |
| 156 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 157 | fn mock_adapter_rate_limit_blocks_burst() { | function signature/body; changes affect API behavior and control-
| 158 |     let temp_allow = std::env::temp_dir().join("allowlist_io_mock_rate_limit.json"); | support logic; impact de
| 159 |     write_test_allowlist(&temp_allow); | protocol or I/O critical; changes may impact external integration corre
| 160 | (blank) | blank separator; no runtime behavior. |
| 161 |     let args = vec![ | support logic; impact depends on neighboring block and data flow. |
| 162 |         "simubench".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 163 |         "--enable-write".to_string(), | protocol or I/O critical; changes may impact external integration corre
| 164 |         format!("--allowlist={}", temp_allow.display()), | support logic; impact depends on neighboring block a
| 165 |         "--noninteractive-approved".to_string(), | support logic; impact depends on neighboring block and data
| 166 |         "--dry-run=false".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 167 |         "--rate-limit-global=1".to_string(), | support logic; impact depends on neighboring block and data flow
| 168 |         "--rate-limit-per-id=1".to_string(), | support logic; impact depends on neighboring block and data flow
| 169 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 170 |     let cfg = HwConfig::from_cli_args(&args).expect("valid args"); | panic boundary; edits alter crash behavior
| 171 | (blank) | blank separator; no runtime behavior. |
| 172 |     let mut ad = MockAdapter::new(); | support logic; impact depends on neighboring block and data flow. |
| 173 |     ad.init(&cfg).expect("init ok"); | panic boundary; edits alter crash behavior and resilience. |
| 174 | (blank) | blank separator; no runtime behavior. |
| 175 |     let f = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
| 176 |         id: 0x18FF50E5, | support logic; impact depends on neighboring block and data flow. |
| 177 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 178 |         data: [0; 8], | support logic; impact depends on neighboring block and data flow. |
| 179 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 180 |         timestamp_ms: None, | support logic; impact depends on neighboring block and data flow. |
| 181 |     }); | support logic; impact depends on neighboring block and data flow. |
| 182 | (blank) | blank separator; no runtime behavior. |
| 183 |     ad.send_frame(f.clone()).expect("first send ok"); | panic boundary; edits alter crash behavior and resilien
| 184 |     let err = ad.send_frame(f).expect_err("second send should rate-limit"); | protocol or I/O critical; changes
| 185 |     assert!(matches!(err, HwError::RateLimited)); | support logic; impact depends on neighboring block and data
| 186 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 187 | (blank) | blank separator; no runtime behavior. |
| 188 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 189 | fn allowlist_parse_mask_accepts_hex() { | function signature/body; changes affect API behavior and control-flow
| 190 |     let mask = io::allowlist::parse_mask("0x1FFFFFFF").expect("mask parse"); | panic boundary; edits alter crash
| 191 |     assert_eq!(mask, 0x1FFF_FFFF); | support logic; impact depends on neighboring block and data flow. |
| 192 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 193 | (blank) | blank separator; no runtime behavior. |
| 194 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 195 | fn replay_can_record_keeps_explicit_timestamp() { | function signature/body; changes affect API behavior and co
| 196 |     let f = Frame::Can(CanFrame { | support logic; impact depends on neighboring block and data flow. |
```

```
| 197 |         id: 0x18FF50E5, | support logic; impact depends on neighboring block and data flow. |
| 198 |         dlc: 8, | support logic; impact depends on neighboring block and data flow. |
| 199 |         data: [1, 2, 3, 4, 5, 6, 7, 8], | support logic; impact depends on neighboring block and data flow. |
| 200 |         len: 8, | support logic; impact depends on neighboring block and data flow. |
| 201 |         timestamp_ms: Some(999), | support logic; impact depends on neighboring block and data flow. |
| 202 |     }); | support logic; impact depends on neighboring block and data flow. |
| 203 | (blank) | blank separator; no runtime behavior. |
| 204 |     let rec = io::replay::frame_to_record(&f, "rx", None, None); | support logic; impact depends on neighboring
| 205 |     assert_eq!(rec.ts, 999); | support logic; impact depends on neighboring block and data flow. |
| 206 |     assert!(rec.raw_hex.is_none()); | support logic; impact depends on neighboring block and data flow. |
| 207 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 208 | (blank) | blank separator; no runtime behavior. |
```

# File Deep Dive: tests/leak_system_integration.rs

Role: automated quality gate and regression coverage

Line count: 84 | Non-empty: 74 | Symbol count: 2

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| is_empty | 5 |
| key_advance | 3 |
| new | 2 |
| e2e_hydraulic_plus_ac_stress_generates_reports | 1 |
| apply_leak_manual_params | 1 |
| tick | 1 |
| predict_leak_scenarios | 1 |
| export_leak_report_json | 1 |
| export_leak_report_csv | 1 |
| export_leak_predictions_json | 1 |
| export_leak_predictions_csv | 1 |
| export_leak_material_catalog_json | 1 |
| export_leak_oil_catalog_csv | 1 |
| monte_carlo_distribution_has_valid_bounds | 1 |
| monte_carlo_leak_predictions | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 4 | fn | e2e_hydraulic_plus_ac_stress_generates_reports | API and behavior coupling to callers/implementers |
| 75 | fn | monte_carlo_distribution_has_valid_bounds | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use auto_breaking::{HeavyMachinery, ManualCircuitParams, OilType}; | import wiring; changing path/name can break
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 4 | fn e2e_hydraulic_plus_ac_stress_generates_reports() { | function signature/body; changes affect API behavior and
| 5 |     let mut bench = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 6 | (blank) | blank separator; no runtime behavior. |
| 7 |     bench.apply_leak_manual_params( | support logic; impact depends on neighboring block and data flow. |
| 8 |         "HYD_MAIN", | support logic; impact depends on neighboring block and data flow. |
| 9 |         ManualCircuitParams { | support logic; impact depends on neighboring block and data flow. |
| 10 |             oil_type: Some(OilType::HydraulicIso68), | support logic; impact depends on neighboring block and da
| 11 |             piston_pressure_bar: Some(210.0), | support logic; impact depends on neighboring block and data flow
| 12 |             operation_pressure_bar: Some(180.0), | support logic; impact depends on neighboring block and data f
| 13 |             pressure_min_bar: Some(45.0), | support logic; impact depends on neighboring block and data flow. |
| 14 |             pressure_mean_bar: Some(155.0), | support logic; impact depends on neighboring block and data flow.
| 15 |             pressure_ideal_bar: Some(175.0), | support logic; impact depends on neighboring block and data flow.
| 16 |             pressure_max_bar: Some(230.0), | support logic; impact depends on neighboring block and data flow. |
| 17 |             pressure_rupture_bar: Some(280.0), | support logic; impact depends on neighboring block and data flo
| 18 |             oring_squeeze_pct: Some(19.0), | support logic; impact depends on neighboring block and data flow. |
| 19 |             compression_set_pct: Some(14.0), | support logic; impact depends on neighboring block and data flow.
| 20 |             base_leak_area_mm2: Some(0.0009), | support logic; impact depends on neighboring block and data flow
| 21 |             max_supported_temp_c: Some(115.0), | support logic; impact depends on neighboring block and data flo
| 22 |         }, | support logic; impact depends on neighboring block and data flow. |
| 23 |     ); | support logic; impact depends on neighboring block and data flow. |
```

```
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 |     bench.bcm.hvac_on = true; | support logic; impact depends on neighboring block and data flow. |
| 26 |     bench.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 27 |     bench.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 28 |     bench.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 29 | (blank) | blank separator; no runtime behavior. |
| 30 |     for i in 0..900 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 31 |         bench.throttle_pct = if i % 120 < 60 { 72.0 } else { 48.0 }; | support logic; impact depends on neighbori
| 32 |         bench.brake_pct = if i % 180 > 150 { 22.0 } else { 0.0 }; | support logic; impact depends on neighboring
| 33 |         bench.loader_lift_cmd = if i % 90 < 45 { 0.8 } else { -0.6 }; | support logic; impact depends on neighbor
| 34 |         bench.loader_tilt_cmd = if i % 70 < 35 { 0.5 } else { -0.4 }; | support logic; impact depends on neighbor
| 35 |         bench.hitch_joystick = if i % 80 < 40 { 0.7 } else { -0.3 }; | support logic; impact depends on neighbori
| 36 |         bench.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 37 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | (blank) | blank separator; no runtime behavior. |
| 39 |     assert!(!bench.leak_reports.is_empty()); | support logic; impact depends on neighboring block and data flow.
| 40 | (blank) | blank separator; no runtime behavior. |
| 41 |     let pred = bench.predict_leak_scenarios(300.0, 0.05); | support logic; impact depends on neighboring block a
| 42 |     assert!(!pred.is_empty()); | support logic; impact depends on neighboring block and data flow. |
| 43 | (blank) | blank separator; no runtime behavior. |
| 44 |     let root = std::env::temp_dir().join("autobreaking_e2e"); | support logic; impact depends on neighboring bloc
| 45 |     bench | support logic; impact depends on neighboring block and data flow. |
| 46 |         .export_leak_report_json(root.join("runtime_report.json")) | support logic; impact depends on neighborin
| 47 |         .expect("runtime json export"); | panic boundary; edits alter crash behavior and resilience. |
| 48 |     bench | support logic; impact depends on neighboring block and data flow. |
| 49 |         .export_leak_report_csv(root.join("runtime_report.csv")) | support logic; impact depends on neighboring b
| 50 |         .expect("runtime csv export"); | panic boundary; edits alter crash behavior and resilience. |
| 51 |     bench | support logic; impact depends on neighboring block and data flow. |
| 52 |         .export_leak_predictions_json(root.join("pred_report.json"), &pred) | support logic; impact depends on ne
| 53 |         .expect("pred json export"); | panic boundary; edits alter crash behavior and resilience. |
| 54 |     bench | support logic; impact depends on neighboring block and data flow. |
| 55 |         .export_leak_predictions_csv(root.join("pred_report.csv"), &pred) | support logic; impact depends on neig
| 56 |         .expect("pred csv export"); | panic boundary; edits alter crash behavior and resilience. |
| 57 |     bench | support logic; impact depends on neighboring block and data flow. |
| 58 |         .export_leak_material_catalog_json(root.join("materials_catalog.json")) | support logic; impact depends o
| 59 |         .expect("materials catalog json export"); | panic boundary; edits alter crash behavior and resilience. |
| 60 |     bench | support logic; impact depends on neighboring block and data flow. |
| 61 |         .export_leak_oil_catalog_csv(root.join("oils_catalog.csv")) | support logic; impact depends on neighborin
| 62 |         .expect("oils catalog csv export"); | panic boundary; edits alter crash behavior and resilience. |
| 63 | (blank) | blank separator; no runtime behavior. |
| 64 |     let hyd = bench | support logic; impact depends on neighboring block and data flow. |
| 65 |         .leak_reports | support logic; impact depends on neighboring block and data flow. |
| 66 |         .iter() | support logic; impact depends on neighboring block and data flow. |
| 67 |         .find(\|r\| r.name == "HYD_MAIN") | support logic; impact depends on neighboring block and data flow. |
| 68 |         .expect("HYD_MAIN report exists"); | panic boundary; edits alter crash behavior and resilience. |
| 69 |     assert!(hyd.recommended_hold_max_bar > hyd.recommended_hold_min_bar); | support logic; impact depends on neig
| 70 |     assert!(!hyd.rca_hint.is_empty()); | support logic; impact depends on neighboring block and data flow. |
| 71 |     assert!(!hyd.pca_recommendation.is_empty()); | support logic; impact depends on neighboring block and data f
| 72 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 73 | (blank) | blank separator; no runtime behavior. |
| 74 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 75 | fn monte_carlo_distribution_has_valid_bounds() { | function signature/body; changes affect API behavior and contr
| 76 |     let bench = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 77 |     let sims = bench.monte_carlo_leak_predictions(40, 180.0, 0.05, 30.0); | support logic; impact depends on neig
| 78 | (blank) | blank separator; no runtime behavior. |
| 79 |     assert!(!sims.is_empty()); | support logic; impact depends on neighboring block and data flow. |
| 80 |     for s in sims.iter().take(20) { | iteration logic; may alter timing, throughput, and safety constraints. |
| 81 |         assert!((0.0..=100.0).contains(&s.final_rupture_probability_pct)); | support logic; impact depends on nei
| 82 |         assert!(s.peak_pressure_bar >= 0.0); | support logic; impact depends on neighboring block and data flow.
| 83 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 84 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: tests/property_invariants.rs

Role: automated quality gate and regression coverage

Line count: 38 | Non-empty: 33 | Symbol count: 2

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| key_advance | 6 |
| new | 2 |
```

```
| tick | 2 |
| physical_levels_stay_within_bounds | 1 |
| can_health_score_is_always_bounded | 1 |
| network_health_score_01 | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 6 | fn | physical_levels_stay_within_bounds | API and behavior coupling to callers/implementers |
| 26 | fn | can_health_score_is_always_bounded | API and behavior coupling to callers/implementers |
```

### Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use auto_breaking::HeavyMachinery; | import wiring; changing path/name can break compilation and module linkage.
| 2 | use proptest::prelude::*; | import wiring; changing path/name can break compilation and module linkage. |
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | proptest! { | support logic; impact depends on neighboring block and data flow. |
| 5 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 6 |     fn physical_levels_stay_within_bounds(throttle in 0.0f64..100.0, brake in 0.0f64..100.0, steps in 1usize..180
| 7 |         let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow.
| 8 |         sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 9 |         sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 10 |         sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 |         for _ in 0..steps { | iteration logic; may alter timing, throughput, and safety constraints. |
| 13 |             sim.throttle_pct = throttle; | support logic; impact depends on neighboring block and data flow. |
| 14 |             sim.brake_pct = brake; | support logic; impact depends on neighboring block and data flow. |
| 15 |             sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 |             prop_assert!((0.0..=100.0).contains(&sim.ecm.fuel_level_pct)); | support logic; impact depends on nei
| 18 |             prop_assert!((0.0..=100.0).contains(&sim.ecm.def_level_pct)); | support logic; impact depends on neig
| 19 |             prop_assert!((0.0..=100.0).contains(&sim.hcm.fluid_level_pct)); | support logic; impact depends on ne
| 20 |             prop_assert!(sim.hcm.system_pressure_bar >= 0.0); | support logic; impact depends on neighboring bloc
| 21 |             prop_assert!(sim.ecm.oil_pressure_kpa >= 0.0); | support logic; impact depends on neighboring block a
| 22 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 24 | (blank) | blank separator; no runtime behavior. |
| 25 |     #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 26 |     fn can_health_score_is_always_bounded(steps in 1usize..240usize) { | function signature/body; changes affect
| 27 |         let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow.
| 28 |         sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 29 |         sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 30 |         sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 |         for _ in 0..steps { | iteration logic; may alter timing, throughput, and safety constraints. |
| 33 |             sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 34 |             let h = sim.can_net.network_health_score_01(); | support logic; impact depends on neighboring block a
| 35 |             prop_assert!((0.0..=1.0).contains(&h)); | support logic; impact depends on neighboring block and data
| 36 |         } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 37 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: tests/speed_regression.rs

Role: automated quality gate and regression coverage

Line count: 66 | Non-empty: 53 | Symbol count: 4

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| bootstrap_running_machine | 4 |
| tick | 4 |
| key_advance | 3 |
| new | 3 |
| set_direction | 1 |
| speed_increases_with_throttle_in_forward | 1 |
| partial_throttle_launch_moves_machine_promptly | 1 |
| brake_reduces_speed_after_acceleration | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 4 | fn | bootstrap_running_machine | API and behavior coupling to callers/implementers |
| 13 | fn | speed_increases_with_throttle_in_forward | API and behavior coupling to callers/implementers |
| 28 | fn | partial_throttle_launch_moves_machine_promptly | API and behavior coupling to callers/implementers |
| 46 | fn | brake_reduces_speed_after_acceleration | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use auto_breaking::ecu_tcm::Direction; | import wiring; changing path/name can break compilation and module linkag
| 2 | use auto_breaking::HeavyMachinery; | import wiring; changing path/name can break compilation and module linkage.
| 3 | (blank) | blank separator; no runtime behavior. |
| 4 | fn bootstrap_running_machine(sim: &mut HeavyMachinery) { | function signature/body; changes affect API behavior an
| 5 |     // OFF -> ACC -> ON -> START/RUN sequence | comment/documentation; changing affects understanding and intent t
| 6 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 7 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 8 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 9 |     sim.tcm.set_direction(Direction::Forward); | support logic; impact depends on neighboring block and data flow
| 10 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 11 | (blank) | blank separator; no runtime behavior. |
| 12 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 13 | fn speed_increases_with_throttle_in_forward() { | function signature/body; changes affect API behavior and contro
| 14 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 15 |     bootstrap_running_machine(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 16 | (blank) | blank separator; no runtime behavior. |
| 17 |     sim.throttle_pct = 55.0; | support logic; impact depends on neighboring block and data flow. |
| 18 |     sim.brake_pct = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 19 | (blank) | blank separator; no runtime behavior. |
| 20 |     for _ in 0..600 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 21 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 22 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 |     assert!(sim.tcm.ground_speed_kmh > 3.0); | support logic; impact depends on neighboring block and data flow.
| 25 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 26 | (blank) | blank separator; no runtime behavior. |
| 27 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 28 | fn partial_throttle_launch_moves_machine_promptly() { | function signature/body; changes affect API behavior and
| 29 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 30 |     bootstrap_running_machine(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 |     sim.throttle_pct = 30.0; | support logic; impact depends on neighboring block and data flow. |
| 33 |     sim.brake_pct = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 34 | (blank) | blank separator; no runtime behavior. |
| 35 |     for _ in 0..240 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 36 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 37 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 38 | (blank) | blank separator; no runtime behavior. |
| 39 |     assert!( | support logic; impact depends on neighboring block and data flow. |
| 40 |         sim.tcm.ground_speed_kmh > 2.0, | support logic; impact depends on neighboring block and data flow. |
| 41 |         "partial throttle should still move the machine visibly from launch" | support logic; impact depends on
| 42 |     ); | support logic; impact depends on neighboring block and data flow. |
| 43 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 44 | (blank) | blank separator; no runtime behavior. |
| 45 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 46 | fn brake_reduces_speed_after_acceleration() { | function signature/body; changes affect API behavior and control
| 47 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 48 |     bootstrap_running_machine(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 |     sim.throttle_pct = 65.0; | support logic; impact depends on neighboring block and data flow. |
| 51 |     sim.brake_pct = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 52 |     for _ in 0..500 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 53 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 54 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 55 |     let v_before = sim.tcm.ground_speed_kmh; | support logic; impact depends on neighboring block and data flow.
| 56 | (blank) | blank separator; no runtime behavior. |
| 57 |     sim.throttle_pct = 0.0; | support logic; impact depends on neighboring block and data flow. |
| 58 |     sim.brake_pct = 70.0; | support logic; impact depends on neighboring block and data flow. |
| 59 |     for _ in 0..250 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 60 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 61 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 62 |     let v_after = sim.tcm.ground_speed_kmh; | support logic; impact depends on neighboring block and data flow.
| 63 | (blank) | blank separator; no runtime behavior. |
```

```
| 64 |     assert!(v_before > 4.0); | support logic; impact depends on neighboring block and data flow. |
| 65 |     assert!(v_after < v_before); | support logic; impact depends on neighboring block and data flow. |
| 66 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

# File Deep Dive: tests/system_failure_scenarios.rs

Role: automated quality gate and regression coverage

Line count: 81 | Non-empty: 65 | Symbol count: 6

## Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| boot_running | 6 |
| new | 5 |
| tick | 5 |
| key_advance | 3 |
| valve_stuck_scenario_keeps_hydraulic_bounds | 1 |
| sensor_bias_scenario_keeps_controller_stable | 1 |
| can_message_loss_fault_transitions_bus_state | 1 |
| inject_bus_off_once | 1 |
| message_storm_scenario_degrades_but_bounds_health | 1 |
| inject_error_once | 1 |
| network_health_score_01 | 1 |
| rapid_setpoint_change_preserves_runtime_safety_bounds | 1 |
```

## Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
|---:|---|---|---|
| 3 | fn | boot_running | API and behavior coupling to callers/implementers |
| 10 | fn | valve_stuck_scenario_keeps_hydraulic_bounds | API and behavior coupling to callers/implementers |
| 25 | fn | sensor_bias_scenario_keeps_controller_stable | API and behavior coupling to callers/implementers |
| 39 | fn | can_message_loss_fault_transitions_bus_state | API and behavior coupling to callers/implementers |
| 51 | fn | message_storm_scenario_degrades_but_bounds_health | API and behavior coupling to callers/implementers |
| 68 | fn | rapid_setpoint_change_preserves_runtime_safety_bounds | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | use auto_breaking::HeavyMachinery; | import wiring; changing path/name can break compilation and module linkage.
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | fn boot_running(sim: &mut HeavyMachinery) { | function signature/body; changes affect API behavior and control-flo
| 4 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 5 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 6 |     sim.key_advance(); | support logic; impact depends on neighboring block and data flow. |
| 7 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 10 | fn valve_stuck_scenario_keeps_hydraulic_bounds() { | function signature/body; changes affect API behavior and con
| 11 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 12 |     boot_running(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 13 | (blank) | blank separator; no runtime behavior. |
| 14 |     sim.loader_lift_cmd = 1.0; | support logic; impact depends on neighboring block and data flow. |
| 15 |     for _ in 0..300 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 16 |         sim.loader_lift_cmd = 1.0; | support logic; impact depends on neighboring block and data flow. |
| 17 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 18 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 19 | (blank) | blank separator; no runtime behavior. |
| 20 |     assert!(sim.hcm.system_pressure_bar >= 0.0); | support logic; impact depends on neighboring block and data fl
| 21 |     assert!((0.0..=100.0).contains(&sim.hcm.fluid_level_pct)); | support logic; impact depends on neighboring blo
| 22 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 23 | (blank) | blank separator; no runtime behavior. |
| 24 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 25 | fn sensor_bias_scenario_keeps_controller_stable() { | function signature/body; changes affect API behavior and co
| 26 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 27 |     boot_running(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 28 | (blank) | blank separator; no runtime behavior. |
| 29 |     for _ in 0..200 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 30 |         sim.ecm.coolant_temp_c += 0.03; // persistent positive bias | support logic; impact depends on neighborin
| 31 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
```

```
| 32 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 33 | (blank) | blank separator; no runtime behavior. |
| 34 |     assert!(sim.ecm.rpm >= 0.0); | support logic; impact depends on neighboring block and data flow. |
| 35 |     assert!(sim.ecm.coolant_temp_c < 140.0); | support logic; impact depends on neighboring block and data flow.
| 36 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 37 | (blank) | blank separator; no runtime behavior. |
| 38 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 39 | fn can_message_loss_fault_transitions_bus_state() { | function signature/body; changes affect API behavior and co
| 40 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 41 |     boot_running(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 42 | (blank) | blank separator; no runtime behavior. |
| 43 |     sim.can_net | support logic; impact depends on neighboring block and data flow. |
| 44 |         .inject_bus_off_once(auto_breaking::BusId::PowertrainHs); | support logic; impact depends on neighboring
| 45 |     sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 46 | (blank) | blank separator; no runtime behavior. |
| 47 |     assert!(sim.can_net.powertrain.state != auto_breaking::CanBusState::ErrorActive); | support logic; impact dep
| 48 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 49 | (blank) | blank separator; no runtime behavior. |
| 50 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 51 | fn message_storm_scenario_degrades_but_bounds_health() { | function signature/body; changes affect API behavior a
| 52 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 53 |     boot_running(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 54 | (blank) | blank separator; no runtime behavior. |
| 55 |     sim.can_net.inject_error_once( | support logic; impact depends on neighboring block and data flow. |
| 56 |         auto_breaking::BusId::PowertrainHs, | support logic; impact depends on neighboring block and data flow.
| 57 |         auto_breaking::CanErrorKind::BablingIdiot, | support logic; impact depends on neighboring block and data
| 58 |     ); | support logic; impact depends on neighboring block and data flow. |
| 59 |     for _ in 0..180 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 60 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 61 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 62 | (blank) | blank separator; no runtime behavior. |
| 63 |     let h = sim.can_net.network_health_score_01(); | support logic; impact depends on neighboring block and data
| 64 |     assert!((0.0..=1.0).contains(&h)); | support logic; impact depends on neighboring block and data flow. |
| 65 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 66 | (blank) | blank separator; no runtime behavior. |
| 67 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 68 | fn rapid_setpoint_change_preserves_runtime_safety_bounds() { | function signature/body; changes affect API behavi
| 69 |     let mut sim = HeavyMachinery::new(); | support logic; impact depends on neighboring block and data flow. |
| 70 |     boot_running(&mut sim); | support logic; impact depends on neighboring block and data flow. |
| 71 | (blank) | blank separator; no runtime behavior. |
| 72 |     for i in 0..500 { | iteration logic; may alter timing, throughput, and safety constraints. |
| 73 |         sim.throttle_pct = if i % 20 < 10 { 90.0 } else { 5.0 }; | support logic; impact depends on neighboring b
| 74 |         sim.brake_pct = if i % 35 < 5 { 35.0 } else { 0.0 }; | support logic; impact depends on neighboring bloc
| 75 |         sim.tick(1.0 / 60.0); | support logic; impact depends on neighboring block and data flow. |
| 76 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 77 | (blank) | blank separator; no runtime behavior. |
| 78 |     assert!((0.0..=100.0).contains(&sim.ecm.fuel_level_pct)); | support logic; impact depends on neighboring bloc
| 79 |     assert!(sim.hcm.system_pressure_bar >= 0.0); | support logic; impact depends on neighboring block and data f
| 80 |     assert!(sim.ecm.oil_pressure_kpa >= 0.0); | support logic; impact depends on neighboring block and data flow
| 81 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## File Deep Dive: tests/vendor_bridge_e2e.rs

Role: automated quality gate and regression coverage

Line count: 81 | Non-empty: 68 | Symbol count: 3

### Function Call Hotspots

```
| Function | Approx Calls In File |
|---|---:|
| push | 3 |
| bin_path | 2 |
| unique_temp_dir | 2 |
| all | 1 |
| cat_comm_bridge_probe_and_live_phases_flash_e2e | 1 |
| from_cli_args | 1 |
| probe_live_adapter | 1 |
| run_all_phases | 1 |
| new | 1 |
```

### Symbol Index

```
| Line | Kind | Name | If Changed, Likely Impact |
```

```
|---:|---|---|---|
| 9 | fn | bin_path | API and behavior coupling to callers/implementers |
| 22 | fn | unique_temp_dir | API and behavior coupling to callers/implementers |
| 33 | fn | cat_comm_bridge_probe_and_live_phases_flash_e2e | API and behavior coupling to callers/implementers |
```

## Line by Line Change Impact

```
| Line | Code | What Happens If This Line Changes |
|---:|---|---|
| 1 | #![cfg(all(target_os = "windows", feature = "vendor-windows"))] | comment/documentation; changing affects understa
| 2 | (blank) | blank separator; no runtime behavior. |
| 3 | use std::fs; | import wiring; changing path/name can break compilation and module linkage. |
| 4 | use std::path::{Path, PathBuf}; | import wiring; changing path/name can break compilation and module linkage. |
| 5 | (blank) | blank separator; no runtime behavior. |
| 6 | use auto_breaking::io::hw::{probe_live_adapter, HwConfig}; | import wiring; changing path/name can break compilati
| 7 | use auto_breaking::io::production_program::{run_all_phases, ProgramOptions}; | import wiring; changing path/name c
| 8 | (blank) | blank separator; no runtime behavior. |
| 9 | fn bin_path(bin_name: &str) -> PathBuf { | function signature/body; changes affect API behavior and control-flow.
| 10 |     let env_key = format!("CARGO_BIN_EXE_{bin_name}"); | support logic; impact depends on neighboring block and c
| 11 |     if let Ok(p) = std::env::var(&env_key) { | runtime gate; condition changes can enable/disable critical paths
| 12 |         return PathBuf::from(p); | support logic; impact depends on neighboring block and data flow. |
| 13 |     } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 14 | (blank) | blank separator; no runtime behavior. |
| 15 |     let mut fallback = PathBuf::from(env!("CARGO_MANIFEST_DIR")); | support logic; impact depends on neighboring
| 16 |     fallback.push("target"); | support logic; impact depends on neighboring block and data flow. |
| 17 |     fallback.push("debug"); | support logic; impact depends on neighboring block and data flow. |
| 18 |     fallback.push(format!("{bin_name}.exe")); | support logic; impact depends on neighboring block and data flow
| 19 |     fallback | support logic; impact depends on neighboring block and data flow. |
| 20 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 21 | (blank) | blank separator; no runtime behavior. |
| 22 | fn unique_temp_dir(prefix: &str) -> PathBuf { | function signature/body; changes affect API behavior and control
| 23 |     let ts = std::time::SystemTime::now() | support logic; impact depends on neighboring block and data flow. |
| 24 |         .duration_since(std::time::UNIX_EPOCH) | support logic; impact depends on neighboring block and data flow
| 25 |         .map(\|d\| d.as_millis()) | support logic; impact depends on neighboring block and data flow. |
| 26 |         .unwrap_or(0); | support logic; impact depends on neighboring block and data flow. |
| 27 |     let p = std::env::temp_dir().join(format!("{prefix}_{ts}")); | support logic; impact depends on neighboring b
| 28 |     fs::create_dir_all(&p).expect("create temp dir"); | panic boundary; edits alter crash behavior and resilience
| 29 |     p | support logic; impact depends on neighboring block and data flow. |
| 30 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
| 31 | (blank) | blank separator; no runtime behavior. |
| 32 | #[test] | comment/documentation; changing affects understanding and intent traceability. |
| 33 | fn cat_comm_bridge_probe_and_live_phases_flash_e2e() { | function signature/body; changes affect API behavior and
| 34 |     let template_dir = unique_temp_dir("autobreaking_cat_template"); | support logic; impact depends on neighbori
| 35 |     let bridge_src = bin_path("cat_comm_bridge"); | support logic; impact depends on neighboring block and data f
| 36 |     assert!(bridge_src.exists(), "cat_comm_bridge binary not found at {}", bridge_src.display()); | support logic
| 37 | (blank) | blank separator; no runtime behavior. |
| 38 |     let bridge_dst = template_dir.join("cat_comm_bridge.exe"); | support logic; impact depends on neighboring blo
| 39 |     fs::copy(&bridge_src, &bridge_dst).expect("copy bridge exe"); | panic boundary; edits alter crash behavior an
| 40 | (blank) | blank separator; no runtime behavior. |
| 41 |     let allowlist_path = template_dir.join("allowlist.json"); | support logic; impact depends on neighboring bloc
| 42 |     fs::write(&allowlist_path, "[]").expect("write allowlist"); | panic boundary; edits alter crash behavior and
| 43 | (blank) | blank separator; no runtime behavior. |
| 44 |     let firmware_path = template_dir.join("firmware.bin"); | support logic; impact depends on neighboring block a
| 45 |     let firmware_payload: Vec<u8> = (0..512).map(\|i\| (i % 251) as u8).collect(); | support logic; impact depend
| 46 |     fs::write(&firmware_path, &firmware_payload).expect("write firmware"); | panic boundary; edits alter crash be
| 47 | (blank) | blank separator; no runtime behavior. |
| 48 |     let args = vec![ | support logic; impact depends on neighboring block and data flow. |
| 49 |         "simulator_cli".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 50 |         "--hw-mode=live".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 51 |         "--vendor-name=cat_comm".to_string(), | support logic; impact depends on neighboring block and data flow
| 52 |         format!("--vendor-template-dir={}", template_dir.display()), | support logic; impact depends on neighbori
| 53 |         "--enable-write".to_string(), | protocol or I/O critical; changes may impact external integration correct
| 54 |         "--noninteractive-approved".to_string(), | support logic; impact depends on neighboring block and data f
| 55 |         "--dry-run=false".to_string(), | support logic; impact depends on neighboring block and data flow. |
| 56 |         format!("--allowlist={}", allowlist_path.display()), | support logic; impact depends on neighboring bloc
| 57 |     ]; | support logic; impact depends on neighboring block and data flow. |
| 58 | (blank) | blank separator; no runtime behavior. |
| 59 |     let cfg = HwConfig::from_cli_args(&args).expect("parse cfg"); | panic boundary; edits alter crash behavior an
| 60 |     let info = probe_live_adapter(&cfg).expect("probe live adapter"); | panic boundary; edits alter crash behavio
| 61 |     assert!(info.contains("bridge_protocol"), "unexpected adapter info: {info}"); | support logic; impact depends
| 62 | (blank) | blank separator; no runtime behavior. |
| 63 |     let report_dir = template_dir.join("reports"); | support logic; impact depends on neighboring block and data
| 64 |     let report_path = run_all_phases( | support logic; impact depends on neighboring block and data flow. |
| 65 |         &cfg, | support logic; impact depends on neighboring block and data flow. |
| 66 |         Some(0x00), | support logic; impact depends on neighboring block and data flow. |
| 67 |         Some(Path::new(&firmware_path)), | support logic; impact depends on neighboring block and data flow. |
```

```
| 68 |         &report_dir, | support logic; impact depends on neighboring block and data flow. |
| 69 |         ProgramOptions { | support logic; impact depends on neighboring block and data flow. |
| 70 |             strict_mode: true, | support logic; impact depends on neighboring block and data flow. |
| 71 |             execute_flash: true, | protocol or I/O critical; changes may impact external integration correctness
| 72 |         }, | support logic; impact depends on neighboring block and data flow. |
| 73 |     ) | support logic; impact depends on neighboring block and data flow. |
| 74 |     .expect("run all phases"); | panic boundary; edits alter crash behavior and resilience. |
| 75 | (blank) | blank separator; no runtime behavior. |
| 76 |     let report = fs::read_to_string(&report_path).expect("read report"); | panic boundary; edits alter crash beha
| 77 |     assert!(report.contains("\"overall_passed\": true"), "report did not pass gate: {report}"); | support logic;
| 78 |     assert!(report.contains("\"conformance_summary\""), "missing conformance summary: {report}"); | support logic
| 79 |     assert!(report.contains("\"service_sid\": \"0x36\""), "missing SID 0x36 evidence: {report}"); | support logic
| 80 |     assert!(report.contains("request_transfer_exit_positive=true"), "missing transfer exit evidence: {report}");
| 81 | } | scope delimiter; structural only, but wrong placement changes ownership/lifetimes. |
```

## Cross File Traceability

- Build root: Cargo.toml -> src/lib.rs, src/main.rs, src/bin/*
- Live flash execution chain: src/bin/simulator_cli.rs -> src/io/production_program.rs -> src/io/live_runner.rs -> src/io/hw.rs -> adapter implementations
- Vendor path: src/io/vendor_cat_comm.rs + src/bin/cat_comm_bridge.rs + tests/vendor_bridge_e2e.rs
- Quality gates: tests/*.rs + .github/workflows/*.yml

## Change Simulation Playbook

1. If you change protocol validation in src/io/live_runner.rs, rerun cargo test --workspace and vendor bridge e2e.
2. If you change adapter contracts in src/io/hw.rs, validate every adapter implementation and cli/prod phase wiring.
3. If you change report schema fields, update consumers, docs, and assertions in tests/vendor_bridge_e2e.rs.
4. If you change Cargo features, verify ci workflows for feature matrix drift.

## Full Raw Context Appendix

This book is machine-generated for maximal practical coverage, including source maps, symbols, call hotspots, and change-impact guidance.