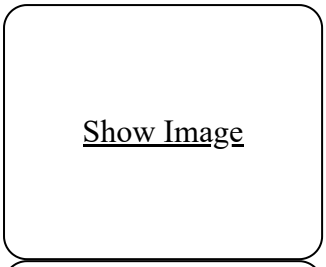
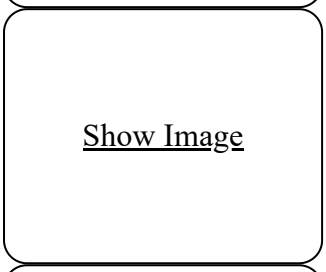
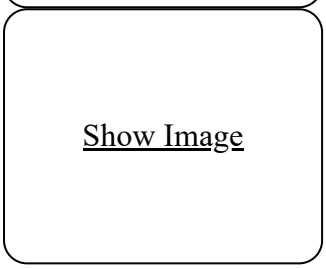
DSE-ASM: Directed Semantic Evolution Assembly Language







An actor-first polyglot assembly language with 100% semantic coherence guarantees.

What is DSE-ASM?

DSE-ASM (Directed Semantic Evolution Assembly Language) is a revolutionary assembly language that combines:

- 🎭 **Actor Model**: Message-passing semantics with error bubbling (not propagation)
- 🔗 **Polyglot Integration**: Seamless interop with C, Python, Rust, Go, Gosilang via GOSSIP protocol
- 🎯 **Semantic Coherence**: 100% coherence maintenance across language/time/space boundaries
- 📡 **REST Observation**: Real-time state monitoring via JSON endpoints
- 🌊 **Directed Evolution**: Programs adapt semantics while preserving meaning

---

Quick Start

bash

```
# Install
git clone https://github.com/obinexus/dse-asm
cd dse-asm/MVP && mkdir build && cd build
cmake -G Ninja .. && ninja && sudo ninja install

# Hello World
echo 'actor main { OBSERVE "init" {}; ret 0 }' > hello.dse
dse-asm compile hello.dse -o hello && ./hello
```

See QUICKSTART.md for detailed examples.

Key Features

1. OBSERVE: Real-Time State Inspection

```
dse

actor main {
    OBSERVE "checkpoint_1" {
        register: rax = 42
        stack_depth: 3
        coherence: 1.0
    }

    ; Your code here

    OBSERVE "checkpoint_2" {}
}
```

2. OBSERVE_REST: Pull JSON State from Endpoints

```
dse

OBSERVE_REST "http://localhost:8080/api/state" {
    method: GET
    poll_interval: 1000  ; ms
    callback: on_state_update
}
```

3. Polyglot GOSSIP: Cross-Language Actors

```
dse
```

```
actor DataProcessor {
    GOSSIP_TO PYTHON {
        module: "ml_model"
        function: "predict"
        args: [data_ptr, size]
    }

    AWAIT result FROM PYTHON
    ret result
}
```

4. Error Bubbling (Not Propagation)

```
dse

actor parent {
    call child
    on_error: { handle_bubbled_error() }
}

actor child {
    if (error) { BUBBLE_ERROR "msg" }  ; Bubbles UP to parent
}
```

Architecture

Integration with OBINexus Ecosystem

```
functor-framework (Type System: He âŠƒ Ho)
```

Toolchain Flow

```
riftlang.exe â†' .so.a â†' rift.exe â†' gosilang
   â†"          â†"        â†"         â†"
  DSE-ASM â†' nlink â†' polybuild â†' executable
```

Core Concepts

Semantic Coherence

DSE-ASM maintains **100% coherence** by tracking semantic drift across:

Dimension	Mechanism	Guarantee
Time	Lossless DAG	O(log n) temporal preservation
Space	Isomorphic DAG	Structure-preserving transforms
Language	GOSSIP protocol	Cross-language semantic validation

He âŠƒ Ho Separation

From **functor-framework**:

```
He (Heterogeneous) âŠƒ Ho (Homogeneous)

DSE-ASM enforces:
- He: Mixed-type actor messages (real-world)
- Ho: Uniform-type optimizations (performance)
- Containment: He always contains Ho
```

Phenomodel Triple

```
Phenotype (Observable State)
   â†" OBSERVE
Phenomemory (Captured Snapshot)
   â†" CONSUME
Phenovalue (Derived Meaning)
```

Build System

CMake Configuration

```
cmake
```

```
cmake_minimum_required(VERSION 3.20)
project(dse-asm VERSION 1.0.0 LANGUAGES C)

# Dependencies
find_package(CURL REQUIRED)
find_package(json-c REQUIRED)

# Build DSE-ASM library
add_library(dse-asm SHARED
    src/core.c
    src/observe.c
    src/observe_rest.c
    src/actor.c
    src/gossip.c
)

target_link_libraries(dse-asm CURL::libcurl json-c::json-c)
```

Build with nlink → polybuild

```
bash

# Generate FFI bindings
nlink generate --from dse-asm --to python,rust,go

# Build entire polyglot stack
polybuild --target dse-asm --config Release

# Run tests
polybuild test --suite integration
```

Examples

Basic Program

```
dse
```

```
; Hello World with OBSERVE
actor main {
    state: isolated;

    OBSERVE "start" {}

    mov rax, [hello_msg]
    call print_string

    OBSERVE "end" {}

    ret 0
}
```

REST Monitoring

```
dse

; Pull sensor data from REST API
actor SensorMonitor {
    OBSERVE_REST "http://iot.local/sensors/temp" {
        poll_interval: 500
        on_update: process_temperature
    }

    fn process_temperature(json: JsonObject) {
        let temp = json.get("celsius")
        if temp > 80 {
            BUBBLE_ERROR "overheating"
        }
    }
}
```

Polyglot ML Pipeline

```
dse
```

```
; DSE-ASM orchestrates Python ML model
actor MLPipeline {
    GOSSIP_TO PYTHON {
        module: "tensorflow_model"
        function: "predict"
        args: [image_data, width, height]
    }

    AWAIT predictions FROM PYTHON

    GOSSIP_TO RUST {
        module: "post_processing"
        function: "apply_filters"
        args: [predictions]
    }

    AWAIT result FROM RUST

    OBSERVE "ml_result" {
        coherence: compute_coherence(result)
    }

    ret result
}
```

See `/examples` for more.

Testing

```bash
# Unit tests
cd build && ctest --output-on-failure

# Integration tests (requires REST mock server)
./tests/integration/run_all.sh

# Polyglot tests
dse-asm test --polyglot python,rust,go

# Coherence validation
dse-asm analyze examples/*.dse --coherence-report
```

OBINexus Constitutional Compliance

DSE-ASM adheres to the **OBINexus Constitutional Framework**:

- **Article II (OpenSense)**: Transparent observation via OBSERVE/OBSERVE_REST
- **Article III (Investment Protection)**: Milestone-based semantic evolution
- **Article V (Human Rights)**: Human-in-loop coherence validation
- **Article VII (#NoGhosting)**: Explicit error bubbling (no silent failures)

See Technical_Specification_-_Gosilang_Design_Infusion_Patents.md for details.

Documentation

- **QUICKSTART.md** - Get started in 5 minutes
- **docs/OBSERVE.md** - OBSERVE instruction reference
- **docs/REST_ADAPTER.md** - OBSERVE_REST configuration
- **docs/GOSSIP_PROTOCOL.md** - Polyglot actor communication
- **docs/ERROR_BUBBLING.md** - Error handling model
- **docs/COHERENCE.md** - Semantic coherence guarantees

Contributing

We welcome contributions that maintain DSE-ASM's core principles:

1. **100% Semantic Coherence**: All changes must pass coherence validation
2. **Error Bubbling**: No downward error propagation
3. **O(log n) Auxiliary Space**: From functor-framework principles
4. **Polyglot Compatibility**: Must work across all supported languages

See CONTRIBUTING.md for guidelines.

Roadmap

v1.0 (Current)

- ✅ Core DSE-ASM language
- ✅ OBSERVE instruction
- ✅ OBSERVE_REST (JSON endpoints)
- ✅ Basic actor model
- ✅ Error bubbling

v1.1 (Q2 2025)

- 🔄 Full GOSSIP protocol (Python, Rust, Go)
- 🔄 Semantic coherence metrics
- 🔄 nlink FFI generation
- 🔄 polybuild integration

v1.2 (Q3 2025)

- ⏳ WebAssembly target
- ⏳ GraphQL OBSERVE endpoints
- ⏳ Real-time coherence dashboard
- ⏳ Quantum-resistant signatures

v2.0 (Q4 2025)

- ⏳ Full BCI integration (opensense-neurospark)
- ⏳ Puppet Protocol relay
- ⏳ Active state machine runtime
- ⏳ Constitutional compliance validator

Related Projects

- **functor-framework** - Type system foundation
- **gosilang** - High-level actor language
- **opensense-neurospark** - BCI application
- **libpolycall** - Polyglot FFI layer
- **hdis** - Hybrid Directed Instruction System

License

Contact

OBINexus Computing

- GitHub: @obinexus
- YouTube: OBINexus Computing

- Website: obinexus.org

"Where assembly meets semantics, actors meet consciousness, and coherence reaches 100%."

Citation

bibtex

```
@software{dse_asm_2025,
  title = {DSE-ASM: Directed Semantic Evolution Assembly Language},
  author = {OBINexus Computing},
  year = {2025},
  url = {https://github.com/obinexus/dse-asm},
  version = {1.0.0}
}
```