Embedded Debugging Techniques

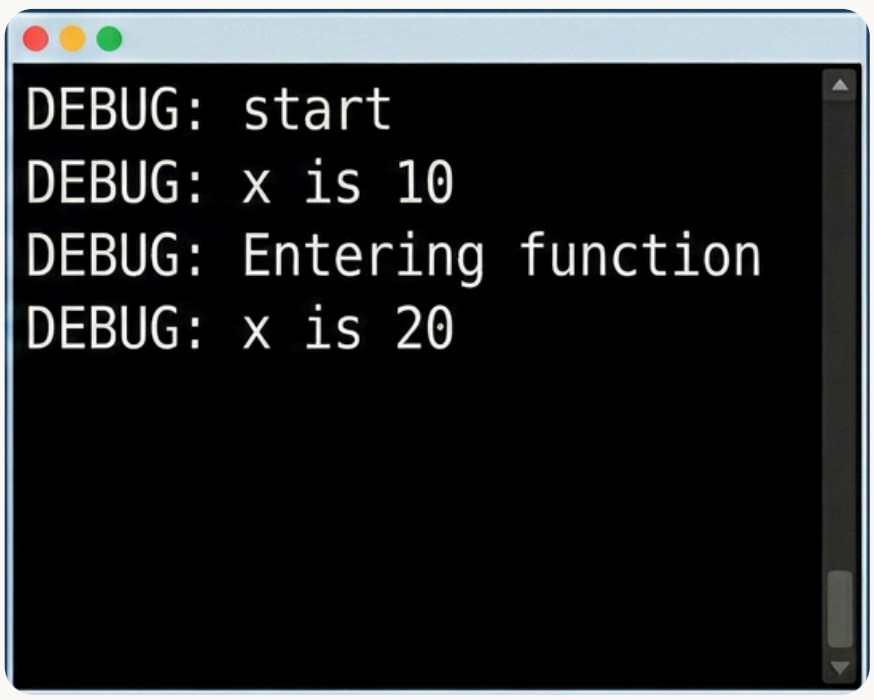
Hunt bugs efficiently!!

Ala Eddine Hammouda

Printf Debugging

Simple. Reliable. Still relevant

- The most primitive, yet universally understood method.
- **How it Works:** Injects code to send status logs or variables to a terminal via UART or RTT.
- **When to Use:** To check program flow or values without halting the CPU.
- **Where it shines**:
  - ✅ Extremely simple
  - ✅ Works without a debugger
  - ✅ Minimal setup required



Ala Eddine Hammouda

GNU Debugger

- **GNU Debugger (GDB)** is a tool for debugging programs in languages like C and C++, allowing developers to inspect and control program execution.

- **What it requires:** a hardware probe like J-Link, ST-Link, or PEmicro to translate high-level software commands into the physical SWD/JTAG signals needed to control a microcontroller.

✅ Best suited for systems that can be safely halted when you need to understand the code behavior.

Ala Eddine Hammouda

Code Execution Control

Single-Stepping:

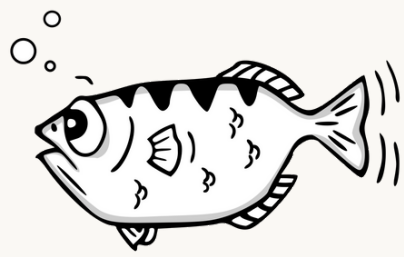
- **Step Over**: Executes the current line without entering any called functions.
- **Step Into**: Executes the current line and enters the called function.
- **Step Out**: Continues execution until the current function returns.

- **Pause/Resume**: Pauses or resumes program execution.
- **Restart**: Restarts the program from the beginning.
- **Stop**: Ends the debugging session.

Breakpoints: Pause program execution when it reaches a specific line or function.

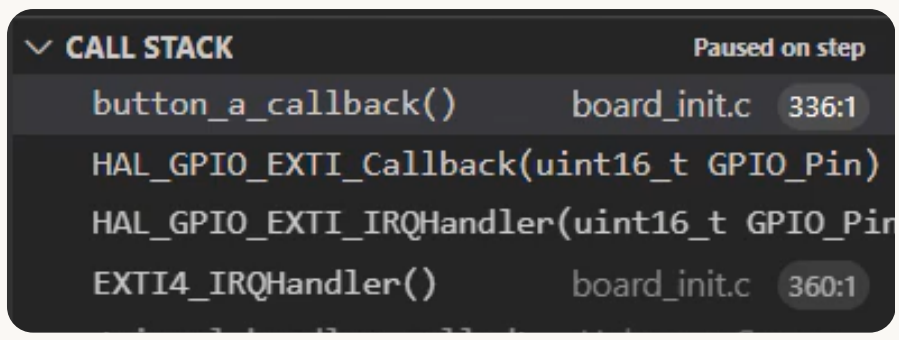
Ala Eddine Hammouda

Inspecting System State

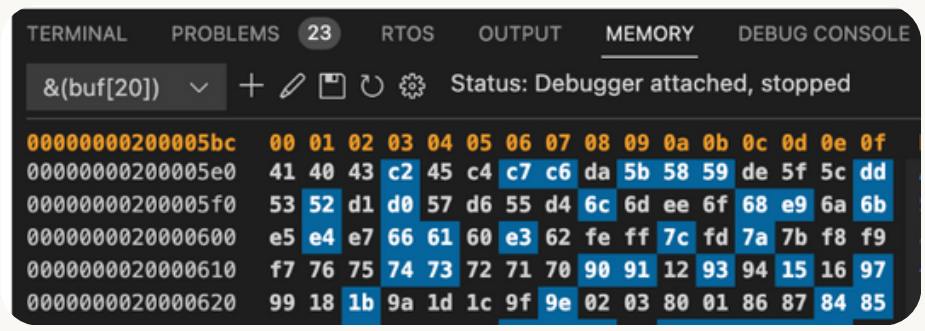


Variable Inspection: View or modify local and global variables.

Backtrace: Display the call stack to see how execution reached the current point.



Register & Memory View: Inspect CPU registers and raw memory addresses.



Live Watch: Monitor variable values in real time.

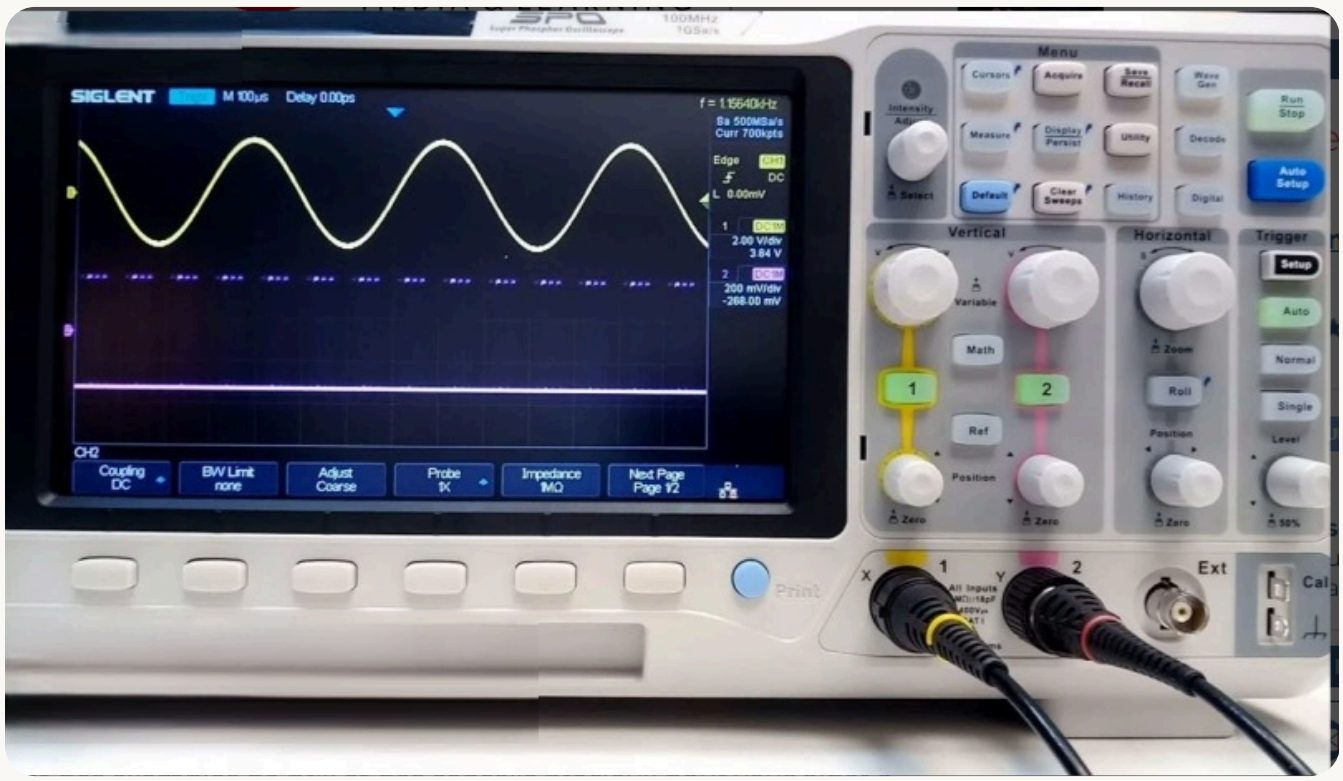
Watchpoints: Halt execution when a variable changes.
Who keeps changing this variable?!
→ A watchpoint will do the job 😄

Ala Eddine Hammouda

Oscilloscopes

How it works: You attach oscilloscope probes to pins (GPIOs, buses, analog lines, etc.), and the scope displays voltage over time, letting you see waveforms, measure timing, and capture glitches.

When to use: Whenever you need to observe what the hardware is actually doing in real time, such as verifying a PWM signal's frequency and duty cycle.



Ala Eddine Hammouda

Logic Analyzers

How it works: Connect a logic analyzer to multiple digital lines (GPIOs, buses like I^2C/SPI, UART TX/RX, etc.). It records the high/low states over time and can decode serial protocols. You then review the capture on a PC to see the exact sequence of events.

What it requires: A hardware interface like a Saleae Logic that samples voltage transitions and translates them into decoded signals.

When to use: When debugging multi-signal timing or digital protocol issues. For example, if an I^2C read fails, a logic analyzer can show whether clock pulses and ACKs occur as expected.

Ala Eddine Hammouda

Memory Profiling

How it works: The memory profiler monitors the application at runtime, recording memory allocations, deallocations, and usage patterns over time.

When to use: If the system crashes or behaves unpredictably after running for a while, a memory profiler can show where memory leaks or stack overflows occur.

Choosing the Right Tool:

- Segger SystemView (RTOS & Bare-Metal)
- Valgrind (Embedded Linux)
- Percepio Tracealyzer (RTOS)

→ These tools provide deep visibility into heap/stack usage, allocation lifetimes, and memory fragmentation.

Ala Eddine Hammouda

👉 Follow for practical embedded insights.

👍 Like & 🔁 Share if you learned something new!

Know other debugging techniques? Share them in the comments! 🙂

Ala Eddine Hammouda