+33 (0)1 48 10 09 46
formation@ambient-it.net
140 Avenue Jean Lolive 93500 Pantin - France

Updated on 11/03/2026

Sign Up

AI Agent Training for Embedded Systems

3 days (21 hours)

Overview

AI-assisted embedded development aligns coding agents with the requirements of the target firmware and hardware. A plausible assumption may produce code that compiles but does not function correctly on the board. An approach based on technical evidence allows you to validate proposals, verify behavior, and justify design choices.

Our AI Agents for Embedded Systems training will enable you to integrate these assistants into your work environment without compromising technical rigor or validation responsibility.

You will learn to formalize verifiable functional specifications, prepare the hardware context, and create appropriate prompts. You will use datasheets, SVD files, and reference manuals to guide the generation of firmware.

By the end of the training, you will be able to configure agents in your IDE and terminal, utilize the Model Context Protocol, and apply specialized skills to orchestrate your tools.

This training also covers on-target testing, static analysis, agent security, and audits. You will compile a validation dossier linking requirements, changes, and measured results to ensure that the firmware remains controlled, tested, and verifiable.

Like all our training courses, this one will introduce you to **the latest stable version** of the technology and its new features.

Objectives

- Structure an AI-assisted embedded development approach based on verifiable evidence
- Write functional specifications and prompts that incorporate hardware constraints
- Configure IDE and CLI tools within a controlled project environment
- Design specialized agents combining MCP, reusable skills, and tool orchestration
- Validate the generated firmware through static analysis, host-based testing, and target
- Audit agent permissions and document the traceability of technical decisions

Target Audience

- Embedded developer
- Firmware engineer
- Embedded systems engineer
- Embedded validation engineer
- Embedded software architect

Requirements

- Experience with embedded programming in C or C++
- Knowledge of microcontrollers, registers, interrupts, and peripherals
- Experience with cross-compilation, debugging, and flashing firmware onto a board
- Proficiency in common version control operations using Git
- Ability to read technical documentation in English and write simple tests

Technical Requirements

- Computer running Windows 11 or Ubuntu 24.04 LTS, with a quad-core processor; 16 GB of RAM recommended and available USB ports; no dedicated graphics card required for hosted assistants
- Installation of Visual Studio Code, Git, and a compatible IDE and CLI assistant, in versions validated for the session
- STM32CubeIDE environment with a working cross-compiler and debugger, according to the configuration provided prior to the training
- STM32 Nucleo board with ST-LINK probe and USB data cable; the specific hardware model specified prior to the session
- Stable Internet connection with authorized access to the necessary services, a GitHub Copilot account with the required features, and local permissions to install the approved tools

Agenda for our “AI Agents for Embedded Systems” training

[Day 1 - Morning]

AI Assistants and Embedded Rigor

- Introduction to the training and the overarching project: device control firmware on a development board
- Overview of AI tools for embedded engineers, conversational assistants, code completion, IDE agents, and CLI agents
- Differences between compilable code, compliance with requirements, and correct behavior on the target hardware
- Principles of embedded prompting, memory constraints, timing, interrupts, and criteria for success
- Evidence-based methodology, separating documented facts, assumptions, and necessary verifications
- Hands-on workshop: Compare two code proposals for a device, identify hardware assumptions, and reformulate a prompt requiring verifiable evidence.

[Day 1 - Afternoon]

Specifications and hardware

context

- Specification-driven development, expected behaviors, interfaces, limitations, and acceptance criteria
- Drafting a functional specification document, states, errors, time limits, and observable
- Using data sheets, reference manuals, board schematics, and manufacturer errata
- Reading CMSIS-SVD files, registers, bit fields, and the limits of their description of hardware behavior
- Establishing the project context, exact component reference, document revisions, existing code, and a record of assumptions
- Hands-on workshop: Drafting the device specification for the overarching project and compiling a context document linking each requirement to its hardware documentation.

[Day 2 - Morning]

Tools in the IDE and terminal

- Configuring IDE and CLI wizards, repository scope, file access, and command approval
- Reusable project instructions, coding conventions, architecture, compilation commands, and validation rules
- Breaking down a request into a change plan, limited tasks, and human checkpoints
- Generation and review of an embedded driver, initialization, registers, data types, memory access, and error handling
  error handling
- Controlled execution of tools, reading Git diffs, and collecting build logs
- Hands-on workshop: Configure a wizard in the IDE and terminal, generate a limited modification to the driver, and verify the changes before compilation.

[Day 2 - Afternoon]

Specialized agents and MCP orchestration

- Model Context Protocol architecture: host, clients, servers, resources, tools, and prompts
- Controlled connection to project resources and compilation or testing tools, with minimal permissions
- Designing a specialized embedded agent: role, inputs, authorized tools, deliverables, and for termination
- Creation of reusable skills, instructions, deterministic scripts, and reporting templates
- Multi-tool orchestration, generation-compilation-testing pipeline, failure handling, and human validation before deployment to the target
- Hands-on workshop: Configure a specialized agent using a prepared local MCP server and a , then verify its permissions and reports on the “fil rouge” project.

[Day 3 - Morning]

Validation of the generated firmware

- Building a requirements and test matrix, measurable criteria, and ensuring the independence of results from the agent’s assertions
- Static analysis and compiler diagnostics, conversions, overflows, memory access, and undefined behaviors
- Unit tests on the host, device emulations, edge cases, and limitations of simulation
- Target testing, debugging, register inspection, traces, and timing measurements
- Identifying hardware discrepancies, volatility, atomicity, concurrency with interrupts, and memory consumption
- Hands-on workshop: Diagnosing firmware that compiles but fails on the board, correcting the cause, and preserve the results of non-regression tests.

[Day 3 - Afternoon]

Security and evidence collection

- AI agent security, prompt injections in documents, untrusted tools, and dangerous commands
- Protecting confidential code, secrets, and project data; selecting transmissible
- Permissions audit, MCP servers, dependencies and configurations, tool isolation, and hardware access control
- Compiling an audit trail, versions, requirements, source materials, changes, and reproducible results
- Acceptance criteria and final human review, requirements coverage, known limitations, and delivery decision
- Hands-on workshop: Audit the end-to-end project workflow, test its safeguards, and present a report justifying the acceptance or rejection of the generated firmware.

Target Audience

This training is intended for both individuals and companies—large or small—seeking to train their teams in new, advanced IT technologies or to acquire specific domain knowledge or modern methodologies.

Assessment upon enrollment

The pre-training assessment complies with Qualiopi quality standards. Upon final registration, the learner receives a self-assessment questionnaire that allows us to evaluate their estimated proficiency level across various types of technologies, as well as their expectations and personal goals regarding the upcoming training, within the limits imposed by the selected format. This questionnaire also allows us to anticipate certain connection or internal security issues within the company (intra-company or virtual classroom) that could pose challenges for monitoring and ensuring the smooth operation of the training session.

Teaching Methods

Hands-On Training: 60% Hands-On, 40% Theory. Training materials distributed in digital format to all participants.

Organization

The course alternates between theoretical instruction from the trainer—supported by examples and reflection sessions—and group work.

Assessment

At the end of the session, a multiple-choice quiz is administered to verify that participants have successfully acquired the skills.

Certification

A certificate will be issued to each trainee who has completed the entire training program.

Training organization registered under number 11 75 54743 75. This registration does not constitute state accreditation.
© Ambient IT 2015-2026. All rights reserved. Paris, France - Switzerland - Belgium - Luxembourg