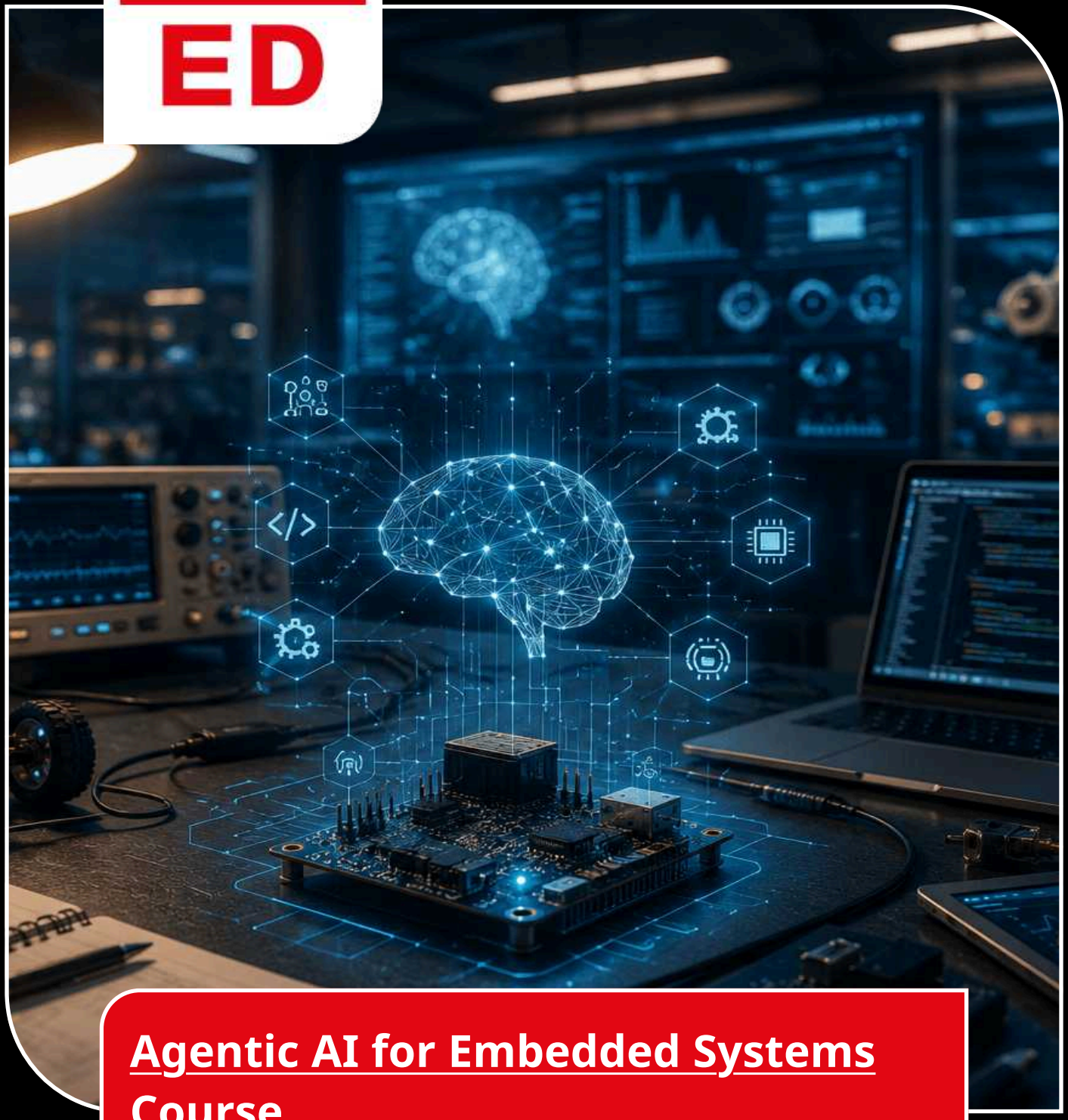
Real Time College

rt-ed.co.il info@rt-ed.co.il 077-7067057



Agentic AI for Embedded Systems Course

Duration: **40** Hours | **Hands-On Training:** 80%

TABLE OF CONTENTS

ABOUT REAL TIME GROUP

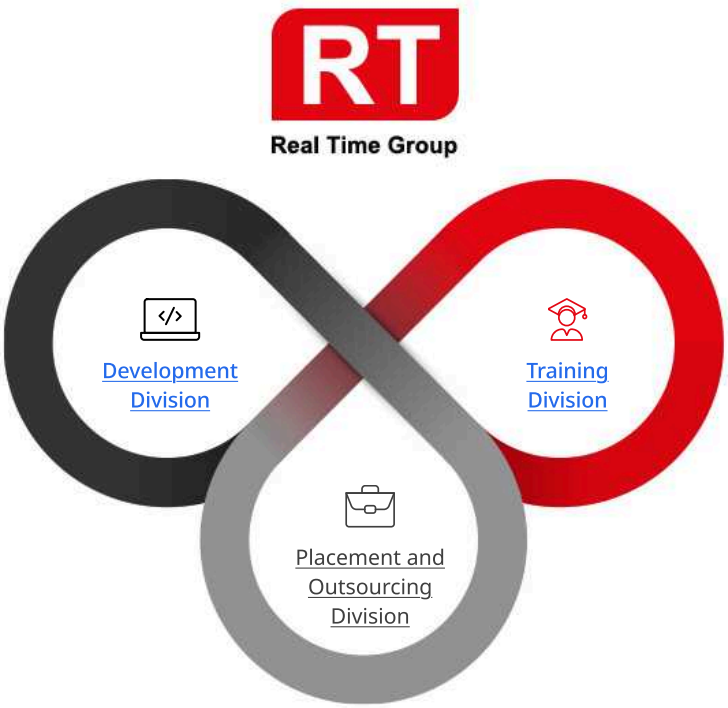
Real Time Group, established in 2007, is a technology company specializing in advanced software and hardware engineering — spanning Embedded Systems, Embedded Linux, FPGAs, AI, Computer Vision, DevSecOps, and Cloud technologies.

Over the past two decades, Real Time teams have delivered complex engineering projects for some of the world's most demanding organizations: Apple, NVIDIA, Intel, the IDF, Rafael, Israel Aerospace Industries, Elbit Systems, and ELTA Systems.

That same depth of real-world expertise drives every course we deliver — so participants learn not just the tools, but how engineers actually use them under pressure.

OUR THREE-DIVISION ECOSYSTEM

Three divisions. One engineering standard.



RT-ED: TRAINING DIVISION

RT-ED delivers advanced professional training built on practical engineering skills. Courses are project-based and use industry-standard development environments, cloud platforms, and modern tooling. Participants gain hands-on experience through structured labs, guided exercises, and industry-grade final projects.

DEVELOPMENT DIVISION

The Development Division delivers engineering and software development services across embedded systems, AI technologies, cybersecurity, and cloud infrastructure. Through direct exposure to real-world workflows and production-grade environments, course participants experience how modern engineering teams actually operate.

PLACEMENT AND OUTSOURCING DIVISION

Real Time Group connects engineers with placement and outsourcing opportunities across leading high-tech organizations. Participants gain access to a professional network built over two decades of hands-on project delivery — opening doors to new roles, consulting engagements, and embedded development projects.

COURSE OVERVIEW

Embedded software development is undergoing its biggest shift in decades. AI agents that read specification documents, reason about real-time constraints, design system architectures, and generate production-ready code are no longer a research curiosity — they are becoming part of the daily engineering workflow.

This program puts that power directly in the embedded engineer's hands. No AI background required. No Python, no machine learning frameworks — just structured, text-based interaction with Claude Code inside the Antigravity IDE, the same way participants will use it on real projects the very next day.

Over five intensive days, engineers go from zero AI tooling experience to commanding a full agentic development pipeline. The central thread is the workflow the industry actually needs: turning a Statement of Work into structured requirements, requirements into a verified architecture, and architecture into disciplined, reviewable embedded code — with the agent doing the heavy lifting and the engineer staying in control.

WHO SHOULD ATTEND\ PREREQUISITES

Who should attend:

- Embedded software and firmware engineers
- Real-time and RTOS developers
- Electronics engineers writing low-level code
- Embedded team leads and software architects
- R&D teams adopting AI-assisted development workflows

Prerequisite:

- Solid C/C++ and embedded experience (RTOS concepts, ISRs, memory constraints)
- Basic familiarity with git and version control
- Comfort working with an IDE and terminal
- No prior AI experience required — the course assumes none

COURSE CURRICULUM

Below is a breakdown of the core modules covered in this course, along with their descriptions and duration.

Select a module to view detailed content.

MODULE NAME	DESCRIPTION
Day 1 — Foundations & Setup	**LLM & Agent Fundamentals** • Tokens, context windows, and forgetting • Hallucination vs. correct output • Chatbots vs. agents: tool use & autonomy • The agent as a supervised junior engineer **Environment Setup** • Installing Antigravity IDE + Claude Code • Permissions model: read, write, execute • Interface tour: chat, terminal, diffs • Smoke test: first guided interaction **Project Context Engineering** • What the agent sees: files, editors, conversation • Writing CLAUDE.md the agent will follow • Repo hygiene: structure, naming, exclusions • Lab: build a CLAUDE.md for an embedded project

MODULE NAME	DESCRIPTION
Day 2 — Prompting & Workflow	**Effective Prompting** • Prompts as specs: goals, constraints, acceptance criteria • Examples and counter-examples • Iterative refinement until results converge • Lab: rewrite weak prompts, compare results **The Agentic Workflow Loop** • Core loop: prompt, plan, act, review, correct • Session management and fresh starts • Git as safety net for agent actions • Lab: full loop on an embedded task **Exploring Codebases** • Mapping unfamiliar repos: structure & data flow • Explaining ISRs and timing-critical paths • Generating call graphs and interface docs • Lab: document a legacy codebase, quiz the results
Day 3 — Context & Requirements	**Content Ingestion** • Token cost and context flooding • Chunking and targeted extraction • Progressive distillation of large documents • Recognizing and recovering from poisoned context **Claude Skills** • How skills are discovered and triggered • Anatomy of SKILL.md • Writing a custom skill for recurring workflows • Lab: create a document-ingestion skill **Requirements Extraction** • Functional, non-functional, constraints • Detecting ambiguities and contradictions • Drafting customer clarification questions • Lab: full extraction on the course example document

MODULE NAME	DESCRIPTION
Day 4 — Architecture & Code	**Architecture Design** • System decomposition: modules, boundaries, ownership • RTOS design: tasks, ISRs, queues, priorities • Interfaces, APIs, timing & memory budgets • Lab: architecture document from Day 3 baseline **Code Generation** • Scaffolding the repo from the architecture • Small, reviewable increments — not big-bang • Enforcing static allocation, HAL, coding standards • Lab: implement selected modules **Verification & Traceability** • Treating agent output as untrusted by default • Agent-assisted code review • Unit tests from the requirements baseline • Requirement → architecture → code traceability
Day 5 — Pipelines & Capstone	**Automating Pipelines** • Custom slash commands for pipeline stages • Subagents and isolated contexts • Plan mode for large multi-step tasks • Lab: wire document-to-code into a pipeline **MCP & External Tools** • What MCP servers add — no coding required • Connecting to docs, issue trackers, analyzers • Permissions and safety with tool access **Capstone: SOW-to-Firmware** • End-to-end run: SOW → requirements → architecture → code • Disciplined checkpoints and review gates • Peer review of generated artifacts

MISCELLANEOUS

- Course opening is contingent upon minimum enrollment.
- Registration fees are non-refundable except in cases where the course does not open.
- All content and terms of this syllabus are subject to Real Time Group's regulations.
- Participants will be notified of any changes to course content or scheduling.
- Real Time Group reserves the right to modify course content at its sole discretion.

Thank You!

Real Time College

rt-ed.co.il

info@rt-ed.co.il

077-7067057