SeekersDen
Embedded systems Training

The Complete Course Syllabus

Our Philosophy: The "Eagle's Tour"

This program is not just a training course; it is a **career trajectory**. The world of embedded systems is vast, and our goal is not to teach you a single, narrow skill. Instead, we provide a high-level, "Eagle's Tour" of the entire modern embedded systems landscape, from the bare metal to the cloud.

Our philosophy is to develop **architects, not just coders**. We provide the foundational knowledge in each key domain—MCUs, RTOS, IoT, FPGAs, and Linux—so you understand how all the pieces fit together. This program provides the essential foundation required to pursue advanced specializations like Embedded AI/ML, Digital Signal Processing (DSP), functional safety, and cybersecurity. By the end of this program, you will have the vision to see the entire landscape and the confidence to go deep on any challenge, putting you on the direct path to becoming a **Full Stack IoT Architect**.

Hardware & Software Tools

Our curriculum is built on a foundation of industry-standard, professional-grade hardware and software. Each of the four modules has its own dedicated hardware kit, ensuring you are learning on relevant, real-world platforms.

- **Module 1 (Foundations & BareMetal):** STM32 Nucleo-64 **(NUCLEO-L476RG)**
- **Module 2 (RTOS & IoT):** STM32 IoT Discovery Kit **(B-L475E-IOT01A)**
- **Module 3 (FPGA Design):** Digilent Arty Spartan-7 **(Arty S7-25)**
- **Module 4 (Embedded Linux):** STM32MP1 Discovery Kit **(STM32MP157C-DK2)**

The primary software used will be professional, free-to-use toolchains like the **STM32CubeIDE** for microcontroller development and **AMD Xilinx Vivado** for FPGA design.

Detailed curriculum

The detailed curriculum for each module is provided on the following pages.

Module 1: Foundations & BareMetal Coding

Goal: To build an unshakable foundation in microcontroller hardware and the essential bare-metal C programming techniques needed to control it.

- **Level 1: Hardware Fundamentals:**
    - Introduction to ARM Architecture and the STM32 Development Environment.
    - Circuit construction, GPIO control for LEDs, and reading switches.
    - Implementing hardware interrupts for efficient input handling.
    - Utilizing peripherals: ADC for analog input, PWM for output, and Timers.
- **Level 2: Communication Protocols:**
    - Parallel Communication – Interfacing with a Character LCD.
    - Asynchronous Communication with UART.
    - Synchronous Communication with SPI.
    - Synchronous Communication with I2C.
- **Level 3: Actuators & Hardware Design:**
    - Controlling DC Motors using H-Bridge drivers.
    - Driving Stepper Motors with precise switching sequences.
    - Introduction to Galvanic Isolation using Relays and Optocouplers.
    - Introduction to Printed Circuit Board (PCB) design concepts.
- **Level 4: Software Fundamentals:**
    - Mastering advanced C concepts like pointers and function pointers.
    - Applying bit manipulation techniques for efficient hardware control.
    - Designing and implementing robust state machines for event-driven systems.
    - Introduction to creating custom, reusable driver libraries.

Module 2: RTOS & IoT

Goal: To learn how to manage complexity using a Real-Time Operating System and to build modern, connected devices.

- **Level 1: RTOS Fundamentals:**
  - Core concepts of Real-Time Operating Systems: tasks, states, and scheduling.
  - Getting started with the FreeRTOS ecosystem and configuration.
  - Creating, deleting, and managing tasks and their priorities.
  - Understanding different scheduling policies and their trade-offs.
- **Level 2: RTOS Communication & Sync:**
  - Inter-task communication and data transfer using Queues.
  - Synchronization between tasks and interrupts using Semaphores.
  - Protecting shared resources with Mutexes.
  - Understanding and solving common concurrency problems like priority inversion.
- **Level 3: IoT & Networking:**
  - Fundamentals of computer networking: IP, TCP, UDP, and sockets.
  - Introduction to common IoT hardware like Wi-Fi and Ethernet modules.
  - Understanding the client-server model for connected devices.
  - Securing embedded devices with basic principles of network security.
- **Level 4: Building a Connected Device:**
  - Introduction to the MQTT protocol for lightweight IoT communication.
  - Sending sensor data from your device to a public MQTT broker.
  - Receiving commands from the cloud to control your device.
  - Architecting a complete sensor-to-cloud IoT application.

Module 3: FPGA Design

Goal: To move beyond programming processors and learn to design custom digital hardware circuits.

- **Level 1: Digital Logic Fundamentals:**
    - Core principles of digital logic: logic gates, Boolean algebra, and number systems.
    - Designing and analyzing combinational logic circuits like adders and multiplexers.
    - Understanding sequential logic circuits including flip-flops, latches, and registers.
    - Introduction to designing and implementing Finite State Machines (FSMs).
- **Level 2: Verilog/VHDL Programming:**
    - Introduction to a Hardware Description Language (Verilog or VHDL).
    - Writing both structural and behavioral code to describe complex digital hardware.
    - Developing comprehensive testbenches to simulate and verify your designs.
    - Learning best practices for writing clean, efficient, and synthesizable HDL code.
- **Level 3: FPGA Implementation:**
    - Understanding the internal architecture of FPGAs and using the Xilinx Vivado toolchain.
    - Mastering the complete design flow: synthesis, place-and-route, and timing analysis.
    - Programming your hardware design onto the Arty S7 board and testing it in real-time.
- **Level 4: System-on-Chip (SoC) Design:**
    - Integrating pre-built Intellectual Property (IP) cores into your design.
    - Interfacing your custom FPGA logic with an embedded soft-core processor.
    - Designing a custom AXI peripheral and connecting it to a system bus.
    - Introduction to High-Level Synthesis (HLS) and advanced on-chip debugging.

Module 4: Embedded Linux

Goal: To master the most powerful platform in embedded systems, preparing for senior-level architectural roles.

- **A Note on Hardware: Why Not a Raspberry Pi?**
  - A Raspberry Pi is a fantastic Single-Board Computer, but it is designed to be easy to use, much like a desktop PC. It abstracts away the complex, low-level details that are the core job of an embedded Linux architect. Our chosen platform, the STM32MP1, is a superior *learning tool* because it forces you to do the real work: configuring bootloaders (U-Boot), writing device trees, and building a custom OS from the ground up. We teach you how to *create* the magic, not just use it.

- **Level 1: Linux for Embedded Systems:**
  - Understanding the Embedded Linux architecture: kernel space vs. user space.
  - Detailed exploration of the Linux boot process on an embedded board.
  - Setting up a professional cross-compilation toolchain.
  - Basic command-line navigation and system administration.

- **Level 2: Device Drivers and BSP:**
  - Writing and debugging basic character device drivers.
  - Understanding the role of the Device Tree in modern Embedded Linux.
  - Configuring and customizing the U-Boot bootloader.
  - Introduction to the Board Support Package (BSP) concept.

- **Level 3: Inter-Process Communication:**
  - Communicating between processes using Pipes and FIFOs.
  - Implementing robust messaging with POSIX and System V Message Queues.
  - High-speed data exchange using Shared Memory.
  - Process synchronization using semaphores and mutexes in user space.

- **Level 4: Advanced Linux Applications:**
  - Introduction to Real-time Linux (PREEMPT_RT).
  - Advanced system debugging and profiling tools (gdb, perf, ftrace).
  - Building a custom, minimal Linux distribution with the Yocto Project.
  - Developing complex, multi-threaded applications in the Linux user space.

For enrollment, please visit ***www.seekersden.in***