**Techno India NJR Institute of Technology, Udaipur**

**Advance Deep Learning Using MATLAB**

**Training Module**

 **Duration: 30 Days**

For an embedded systems training module focused on the **8051 Microcontroller** and the **STM32 (ARM Cortex) Controller**, you can organize it into sections that cover both theoretical concepts and practical, hands-on exercises. Here's a proposed structure for the training:

**Module 1: Introduction to Embedded Systems**

1. **What is an Embedded System?**
	* Definition and Characteristics
	* Real-life Applications
	* Overview of Microcontrollers and Microprocessors
2. **Basics of Embedded Systems Programming**
	* Role of Firmware
	* High-level vs. Low-level programming
	* Cross-compilers and Integrated Development Environments (IDEs)

**Module 2: 8051 Microcontroller Basics**

1. **Introduction to 8051 Architecture**
	* History of 8051
	* Basic Architecture (ALU, Registers, Timers, Serial Port, Interrupts)
	* Pin configuration
	* Addressing modes (Immediate, Direct, Indirect, Indexed)
2. **Assembly and C Programming for 8051**
	* Writing a simple "Hello World" Program (LED blinking)
	* Concepts of Embedded C for 8051
	* Using Keil uVision IDE
3. **Peripherals and Interfacing**
	* Timers and Counters
	* Interrupts Handling
	* Serial Communication (UART)
	* Interfacing with Sensors and Actuators (LEDs, Switches, LCD Display)

**Hands-on:**

* LED Blinking, Push Button Interfacing
* Serial Communication with 8051 (Sending data to PC via UART)
* LCD Interfacing Project

**Module 3: STM32 (ARM Cortex) Microcontroller Basics**

1. **Introduction to STM32 Controllers**
	* Overview of ARM Cortex-M Series
	* STM32 Architecture (System, Bus Architecture, Clocks)
	* Pinout and Peripherals (GPIO, Timers, UART, SPI, I2C)
2. **Development Environment Setup**
	* Installing STM32CubeIDE
	* Getting started with STM32CubeMX (Code Generator for Peripheral Configuration)
3. **Programming STM32 with C/C++**
	* Writing a Basic Program (LED Blinking)
	* Interrupt Handling and Timer Configurations
4. **Advanced Peripherals**
	* ADC (Analog to Digital Converter) and DAC (Digital to Analog Converter)
	* Communication Protocols: SPI, I2C, UART
	* PWM (Pulse Width Modulation) for motor control

**Hands-on:**

* Basic GPIO (LED blinking, Button interfacing)
* UART Communication (Send/Receive Data)
* ADC Reading from Analog Sensors
* PWM-based Motor Control

**Module 4: Comparative Analysis and Applications**

1. **8051 vs STM32**
	* Key Differences (Performance, Power Consumption, Peripheral Set)
	* Use Cases for 8051 and STM32
2. **Real-world Applications**
	* Home Automation (8051)
	* IoT Systems (STM32)
	* Robotics (Interfacing Motors and Sensors)

**Hands-on Final Project:**

* Design a basic Home Automation System using either 8051 or STM32.
* Interfacing sensors like Temperature, Motion Detection, and controlling devices via UART, I2C, or SPI.

**Module 5: Debugging and Optimization Techniques**

1. **Common Issues in Embedded Systems**
	* Debugging tools and techniques for 8051 and STM32
	* Optimizing Power Consumption for Low-Power Devices
2. **Best Practices**
	* Code Optimization for Memory and Performance
	* Use of Interrupts and Timers for Efficient Processing