Baremetal programming involves directly interacting with the microcontroller's hardware, without the use of high-level abstractions such as hardware abstraction layers (HAL) or operating systems like FreeRTOS. In STM32 baremetal programming, you work closely with the microcontroller's registers to control peripherals and write efficient code. This gives you full control over the hardware, allowing for high performance and precision, but requires a deeper understanding of the microcontroller's architecture.

Here is a structured overview of what learning **STM32 baremetal programming** typically involves:

Module 5: EMBEDDED (STM 32 – Baremetal) - 2 Hours / Day

DAYS	PARTICULARS
Day 1	STM32 Basic Overview – STM32F, – OSI Layers and Applications (Advance than other controller), STM32CubeMX CONFIG, STM32CUBEIDE Programming and Debugging
Day 2	STM 32 Internal Architecture and Interface Protocol
Day 3	STM32 GPIO Configuration (LED,Switch,7-segment,LCD,Matrix keypad)
Day 4	ADC (Sensor interfacing)
Day 5	Timer and its interrupts
Day 5	CCP module(Capture, Compare, PWM)
Day 7	DAC
Day 8	Communication protocols(UART,I2C,SPI)
Day 9	I2C With OLED
Day 10	SPI with EEPROM
Day 11	CAN protocol implementation
Day 12	Bootloader overview
Day 13	Blue tooth module, GSM module interfacing
Day 14	GPS,Lora module interfacing
Day 15	RFID interfacing, Motor control
Day 16-Day 25	Project Work Prototype

Outcome of Learning STM32 Baremetal Programming

Upon completing this learning path, you will:

- Have a deep understanding of STM32 microcontroller architecture and peripheral handling.
- Be capable of configuring and controlling peripherals manually by writing directly to the hardware registers.
- Understand and implement interrupts, timers, communication protocols (UART, SPI, I2C) without any abstraction layers.
- Be able to write highly optimized, low-level code that is efficient and precise for embedded systems applications.

This approach gives you full control over the microcontroller hardware and teaches you to write highly efficient, resource-conscious applications.