**STM32CubeIDE**

1. **What Is STM32CubeIDE?**

STM32CubeIDE is an official IDE from STMicroelectronics used to develop applications for STM32 microcontrollers. It includes:

* Code editor + compiler (based on Eclipse & GCC)
* GUI for pin configuration (STM32CubeMX integrated)
* Debugger via ST-LINK
* Supports **HAL** and **LL (Low Layer)** drivers

1. What Is HAL (Hardware Abstraction Layer)?

**HAL is a library provided by ST** to abstract hardware-level programming:

* Instead of writing low-level register code, you use **HAL functions**.

For example, instead of setting GPIO registers manually, you just call:

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_5, GPIO\_PIN\_SET);

This makes your code:

* Easier to write
* More readable
* **Portable** across STM32 MCUs

1. Basic Project Flow in STM32CubeIDE :

**Step-by-Step: First Project** (e.g., LED Blink on STM32F401RE)

Step 1: Create a New STM32 Project

* Open **STM32CubeIDE**
* Click **File > New > STM32 Project**
* Search for **STM32F401RE** or **STM32H753ZI**
* Click **Next**, give a name like LED\_Blink, click **Finish**

Step 2: Pin Configuration (Using CubeMX GUI)

* In **Pinout & Configuration**, click on the pin (e.g., **PA5** — onboard LED)
* Set it to **GPIO\_Output**
* Go to **Clock Configuration** tab (usually auto-set for basic projects)
* Go to **Project Manager > Code Generator**
* Keep it set to **HAL drivers**
* Click **GENERATE CODE**

Step 3: Write Your Code in main.c

int main(void) {

HAL\_Init();

SystemClock\_Config();

MX\_GPIO\_Init();

while (1) {

HAL\_GPIO\_TogglePin(GPIOA, GPIO\_PIN\_5); // Toggle LED

HAL\_Delay(500); // Delay 500ms

}

}

**Step-by-Step Meaning of Each Line:**

|  |  |
| --- | --- |
| |  |  | | --- | --- | | **Code** | **Purpose** | | HAL\_Init(); | Initializes HAL drivers | | SystemClock\_Config(); | Sets up system clock (based on CubeMX config) | | MX\_GPIO\_Init(); | Initializes GPIO pins (PA5 in this case) | | HAL\_GPIO\_TogglePin(); | HAL function to toggle GPIO | | HAL\_Delay(); | Creates a blocking delay in milliseconds |   **Core Concepts You Need to Learn:**   |  |  | | --- | --- | | **Topic** | **What to Learn** | | **GPIO** | Digital Input/Output (LED, Button) | | **Timers** | PWM generation, delay without HAL\_Delay() | | **USART (UART)** | Serial communication | | **I2C/SPI** | Communicate with sensors, displays | | **ADC** | Read analog values (LM35, potentiometer) | | **EXTI (Interrupts)** | Button press without polling | | **DMA** | Efficient data transfer without CPU | | **RTOS (FreeRTOS)** | Multitasking in embedded apps (later) | |  |

**Recommended Practice Projects (Progress Order)**

|  |  |  |
| --- | --- | --- |
| **#** | **Project** | **Concepts** |
| 1 | LED Blink | GPIO Output, HAL\_Delay |
| 2 | Button Toggle LED | GPIO Input, Debounce, EXTI |
| 3 | UART Send/Receive | USART, HAL\_UART\_Transmit |
| 4 | ADC Read Temp Sensor | ADC, HAL\_ADC\_PollForConversion |
| 5 | OLED Display | I2C, SSD1306 |
| 6 | SD Card Logger | SPI, FatFs, File I/O |
| 7 | Sensor Data to Cloud | UART + ESP8266 WiFi |
| 8 | FreeRTOS LED Tasks | RTOS, Threads, Task Delay |