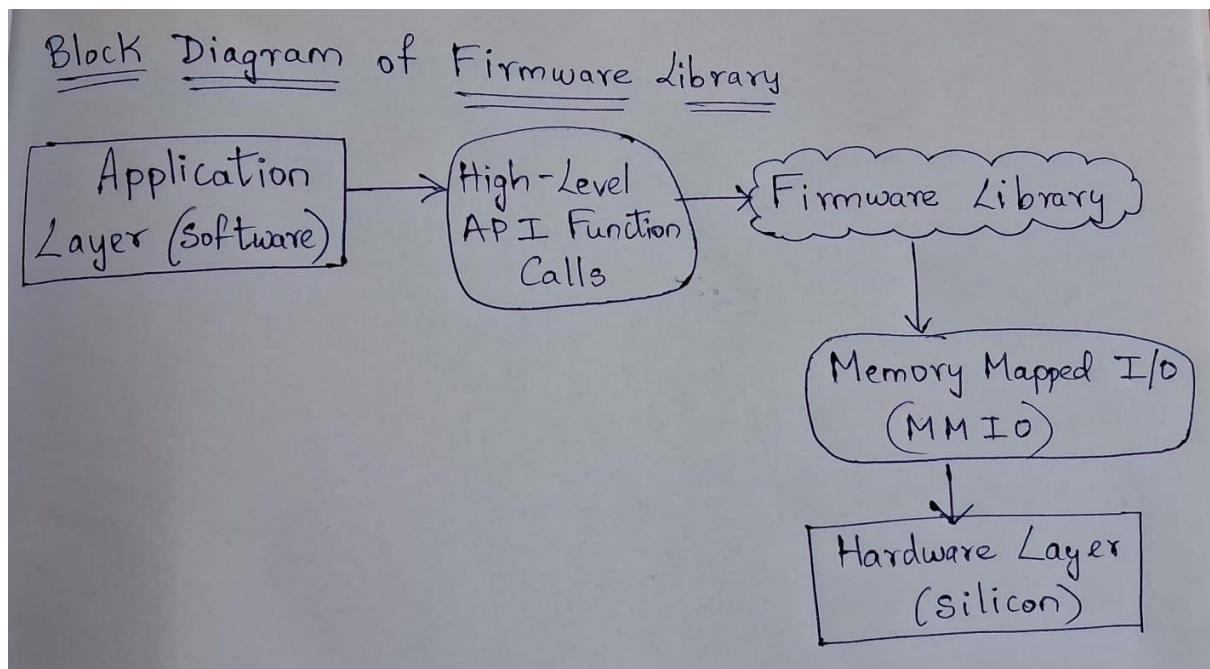


## RISC-V Embedded Firmware Internship

### Task 1: Firmware Foundations & Environment Setup

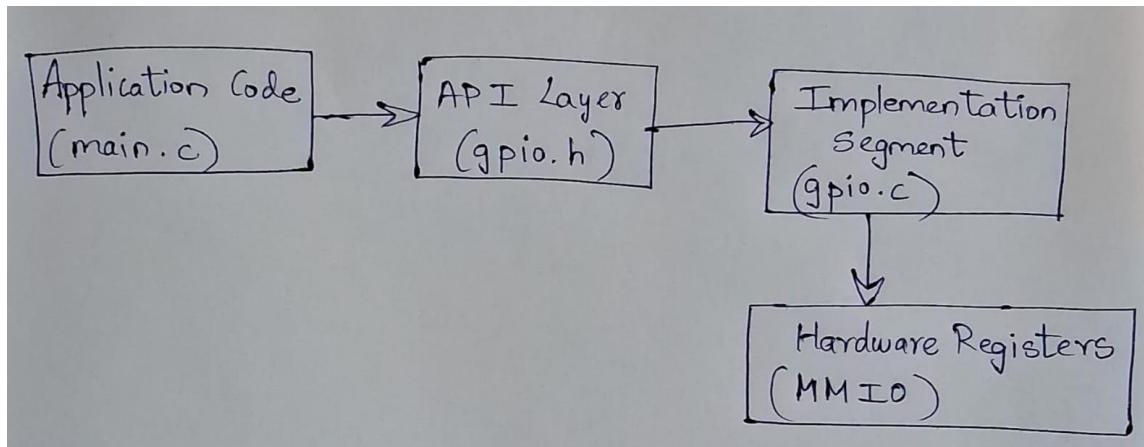
#### 1. What is a firmware library?

- Firmware is the bridge between physical hardware and software logic.  
Firmware manages low-level logic and sensory inputs which is stored in non-volatile memory such as flash/ROM.
- Firmware library is a structured collection of C functions which has direct control over the hardware register access through fixed memory addresses.



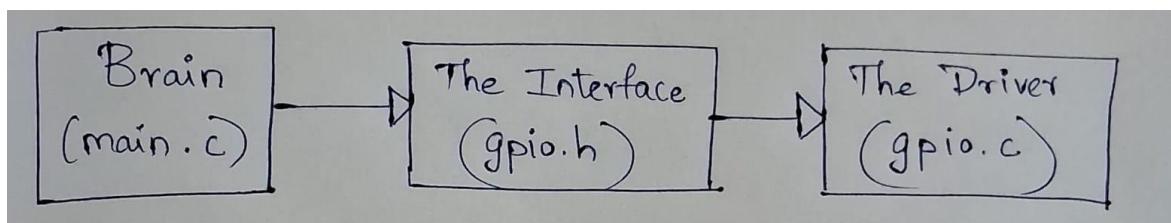
## 2. Why APIs are important in embedded systems?

- APIs are necessary in embedded systems because it can manage the interaction between various software and hardware components.
- The functions like digitalWrite() and delay() helps in register management instead of writing directly to the memory addresses(registers).



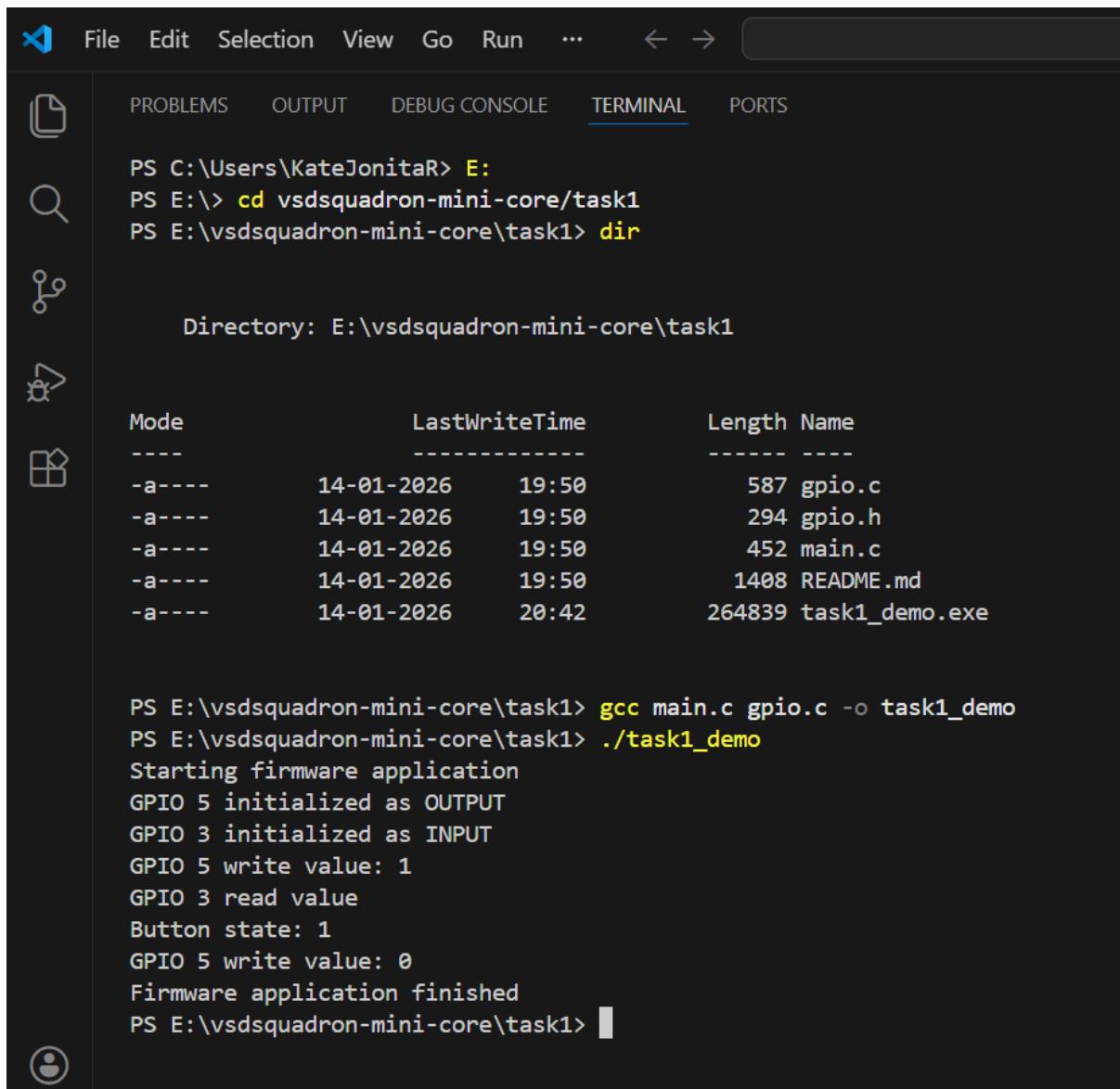
## 3. What was understood from the lab code?

- The code can be organized into layers for easy management and for a smoother execution of the task.
- main.c is used for function calling and implements system logic.
- gpio.h defines the interface and declares what functions exist.
- gpio.c performs register-level work.



## Lab – Understanding a simple firmware library

Console logs showing simulated GPIO behaviour



The screenshot shows a terminal window within a dark-themed IDE interface. The terminal tab is active, displaying the following command-line session:

```
PS C:\Users\KateJonitaR> E:  
PS E:\> cd vsdsquadron-mini-core/task1  
PS E:\vsdsquadron-mini-core\task1> dir  
  
Directory: E:\vsdsquadron-mini-core\task1  
  
Mode                LastWriteTime       Length Name  
----                -----          -----  
-a----    14-01-2026     19:50            587  gpio.c  
-a----    14-01-2026     19:50            294  gpio.h  
-a----    14-01-2026     19:50           452  main.c  
-a----    14-01-2026     19:50          1408 README.md  
-a----    14-01-2026    20:42        264839 task1_demo.exe  
  
PS E:\vsdsquadron-mini-core\task1> gcc main.c gpio.c -o task1_demo  
PS E:\vsdsquadron-mini-core\task1> ./task1_demo  
Starting firmware application  
GPIO 5 initialized as OUTPUT  
GPIO 3 initialized as INPUT  
GPIO 5 write value: 1  
GPIO 3 read value  
Button state: 1  
GPIO 5 write value: 0  
Firmware application finished  
PS E:\vsdsquadron-mini-core\task1>
```

```
PS E:\vsdsquadron-mini-core\task1> Executing by KateJonita> gcc main.c gpio.c -o task1_demo  
PS E:\vsdsquadron-mini-core\task1> Executing by KateJonitaR ./task1_demo  
Starting firmware application  
GPIO 5 initialized as OUTPUT  
GPIO 3 initialized as INPUT  
GPIO 5 write value: 1  
GPIO 3 read value  
Button state: 1  
GPIO 5 write value: 0  
Firmware application finished  
PS E:\vsdsquadron-mini-core\task1> Executing by KateJonitaR
```