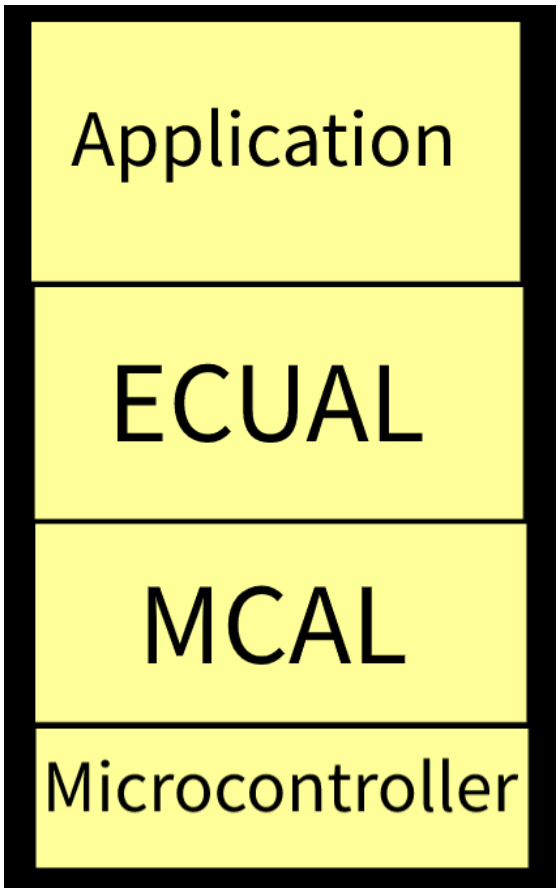
Layered architecture



Layers description:

(1)- Application layer:

Contains functions calls to implement the main project.

(2)- ECUAL: “Electronics Unit Abstraction Layer”

Contains Drivers of the external electronic devices which will be connected to the microcontroller and the system overall.

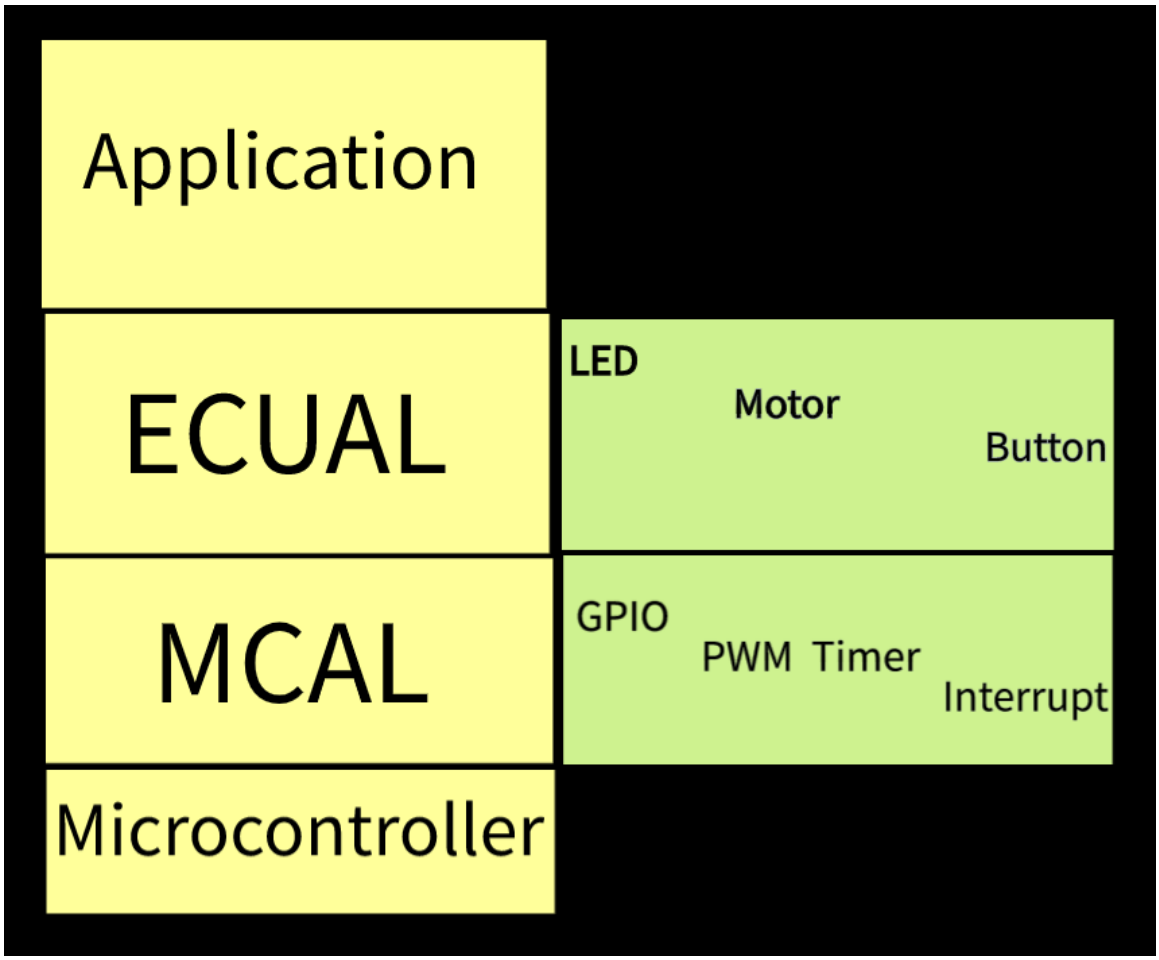
(3)-MCAL: “Microcontroller Abstraction Layer”

Contains interfaces of the microcontroller’s peripherals.

(4)-Microcontroller:

The microcontroller type that will be used to implement the project.

System modules



APIs

1- GPIO API:

Functions prototypes:

```
void DIO_init(uint8_t pinNumber, uint8_t portNumber, uint8_t direction);

void DIO_write(uint8_t pinNumber, uint8_t portNumber, uint8_t value);

void DIO_toggle(uint8_t pinNumber, uint8_t portNumber);

void DIO_read(uint8_t pinNumber, uint8_t portNumber, uint8_t *value);
```

2- Interrupt API:

Functions prototypes:

#define sei() __asm__ __volatile__ (“sei” ::: “memory”)

#define cli() __asm__ __volatile__ (“cli” ::: “memory”)

#define ISR(INT_VEC) void INT_VECT(void) __attribute__ ((signal,used));\

void INT_VECT (void)

3- LED API:

Functions prototypes:

```
void LED_init(uint8_t ledPort, uint8_t ledPin);

void LED_on(uint8_t ledPort, uint8_t ledPin);

void LED_off(uint8_t ledPort, uint8_t ledPin);

void LED_toggle(uint8_t ledPort, uint8_t ledPin);
```

4- Motor API:

Functions prototypes:

```
Std_ReturnType dc_motor_init(const dc_motor_t *_dc_motor);

std_ReturnType dc_motor_right(const dc_motor_t *_dc_motor);
std_ReturnType dc_motor_stop(const dc_motor_t *_dc_motor);
```

5- Button API:

Functions prototypes:

```
void Button_init(uint8_t buttonPort, uint8_t buttonPin);

void Button_read(uint8_t buttonPort, uint8_t buttonPin, uint8_t *value);
```

6-PWM API:

Functions prototypes:

7-Timer API:

Functions prototypes: