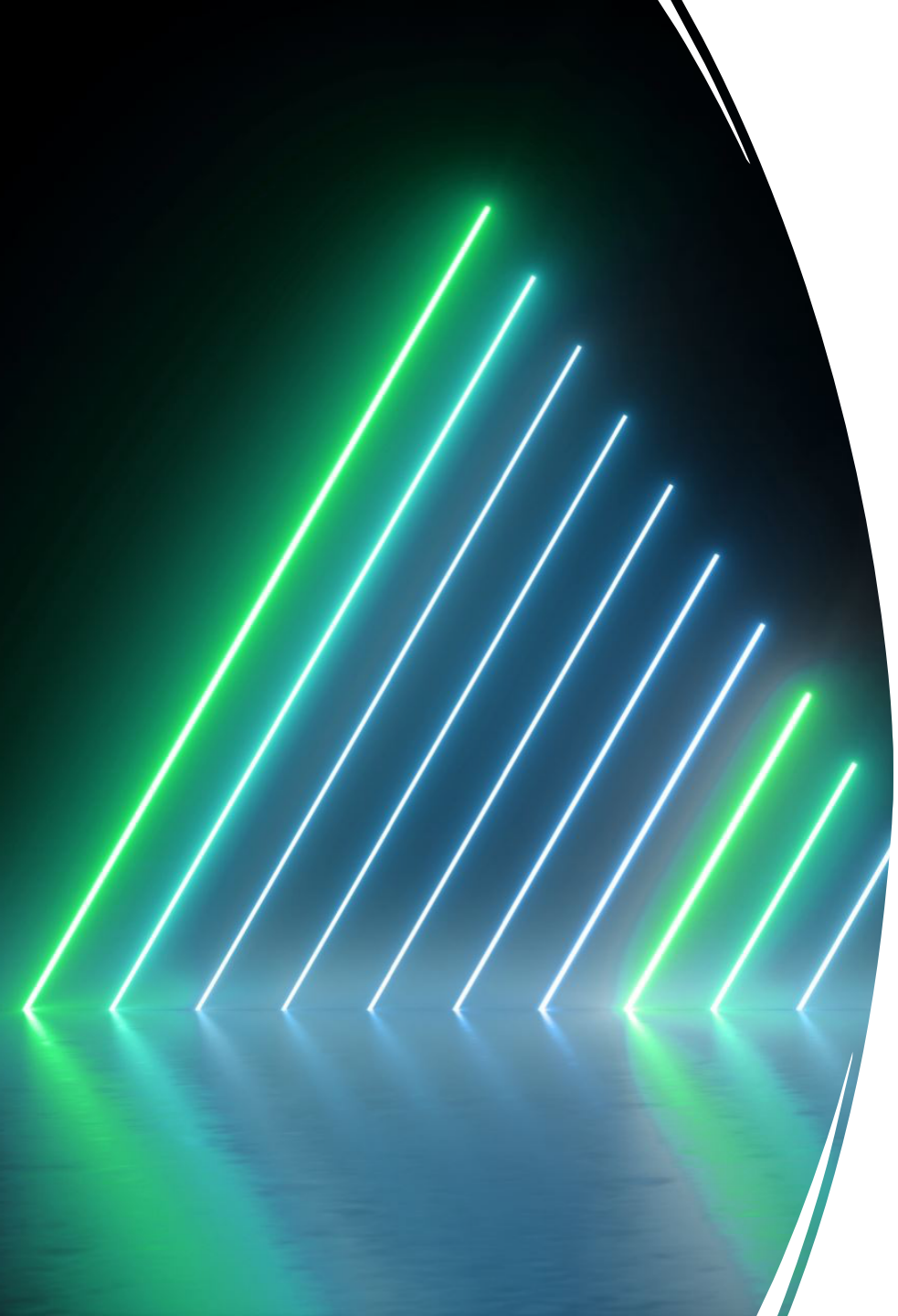


FINAL PROJECT - CAR CONTROL

Static Design for Car Control

HARDWARE USED

TWO MOTORS

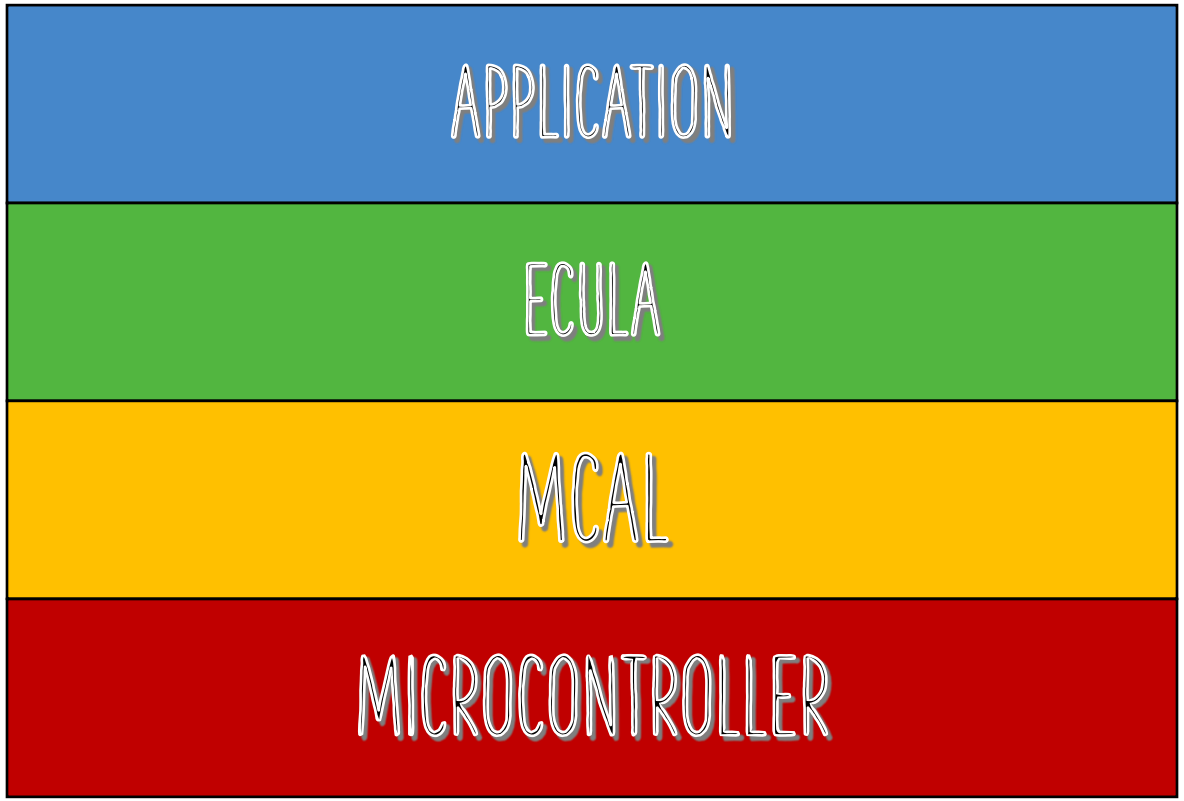
FOUR PUSH BUTTONS

MICROCONTROLLER

LAYERED ARCHITECTURE

The system will be divided into 4 layers

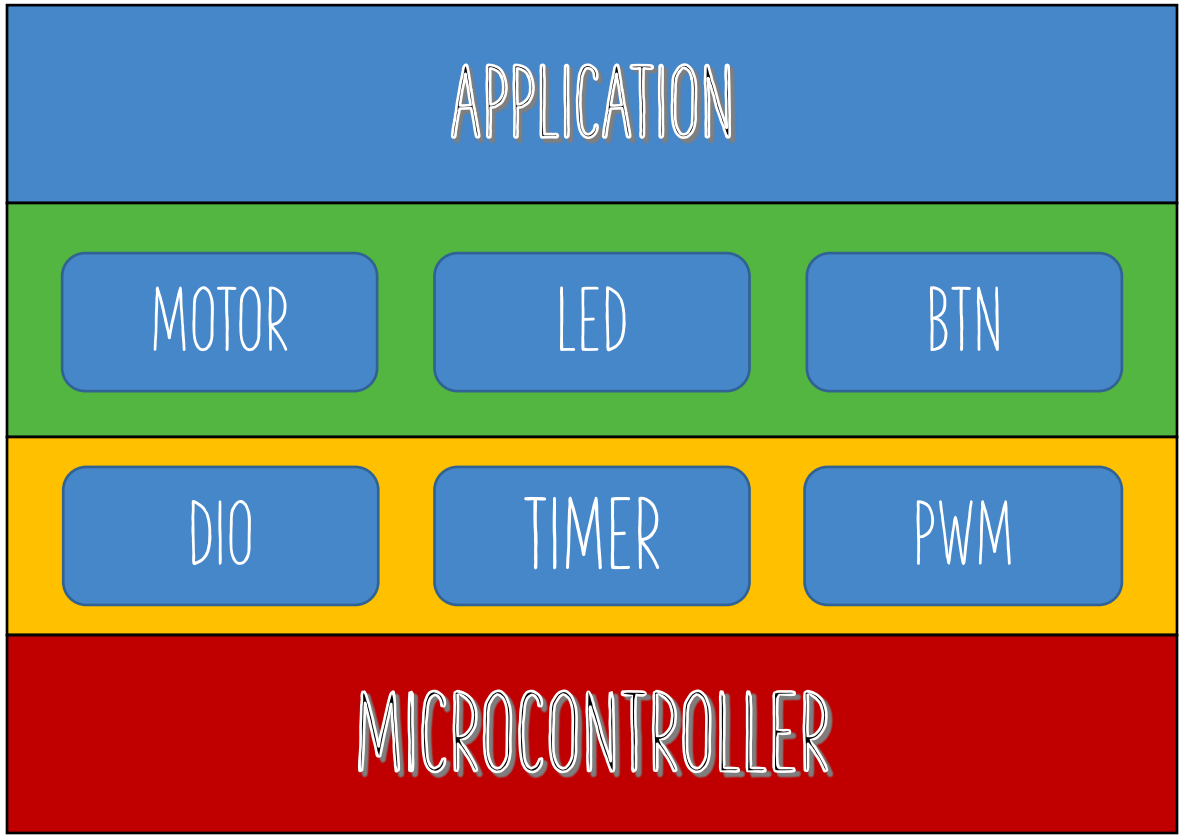
- Application
- ECULA
- MCAL
- Microcontroller



MODULAR PROGRAMMING

Layers will be divided into small units to perform a certain function, this units are called modules.

- ECULA
  - Four Buttons
  - Two Motors
- MCAL
  - DIO
  - Timer
  - PWM



MODULES APIS (FUNCTIONS PROTOTYPES)

TIMER

```
void timerInit(EN_TIMERMODE_t mode, EN_CLOCKSOURCE_t source); //initialize timer with its mode (normal, CTC, fast PWM, PWM Phase Correct) and prescaler
void setCompare(uint8_t cmpValue,EN_CMPMODE_t cmpMode);
//Set compare value for CTC, fast PWM, PWM Phase Correct and its mode on OC0 pin
void timerdelayMS(int32_t timeMS); //Delay time in millisecond
void timerdelay(int32_t time); //Delay time given prescaler (noclk,clk,clk_8,clk_64,clk_256,clk_1024,clk_ext_rising,clk_ext_falling)
void prescaler(EN_CLOCKSOURCE_t source);
void timerStop(); //Private  function don't call
void timerCycle(); //private function don't call
```

PWM

```
void SoftPWM_init(uint8_t output_port, uint8_t output_pin);
//init of timer 0 in normal mode with prescalar 64 + init of output pin
void SoftPWM_pulse(uint8_t dutycycle, uint8_t output_port, uint8_t output_pin);
//generates one pulse on, one off
void SoftPWM_pulse(uint8_t dutycycle, uint8_t* output_ports, uint8_t* output_pins, uint8_t num_of_pins);
//same as previous but simultaneously for multiple pins
```

MODULES APIS (FUNCTIONS PROTOTYPES)

DIO

```
void DIO_init(uint8_t port, uint8_t pin, EN_bit data, EN_bit ctrl);//initialize the data and control of the pin
uint8_t DIO_read(uint8_t port, uint8_t pin); //reads pin value and return 0 for low otherwise high
void DIO_set (uint8_t port, uint8_t pin, EN_bit value); //set the ctrl value on a pin
void DIO_write (uint8_t port, uint8_t pin, EN_bit value); //write data on a pin high or low
void DIO_toggle(uint8_t port, uint8_t pin); //toggle data on a pin
// DIO bit manipulation
uint8_t readbit ( volatile uint8_t* portadd, uint8_t bit);
void setbit (volatile uint8_t* portadd, uint8_t bit);
void tglbit (volatile uint8_t* portadd, uint8_t bit);
void clrbit (volatile uint8_t* portadd, uint8_t bit);
```

MODULES APIS (FUNCTIONS PROTOTYPES)

ECUAL Modules

LED

```
void LED_ON(uint8_t port, uint8_t pin);
void LED_OFF(uint8_t port, uint8_t pin);
void LED_toggle(uint8_t port, uint8_t pin);
void LED_init(uint8_t port, uint8_t pin, EN_bit data);
```

BTN

```
void Button_init(uint8_t port_num, uint8_t pin_num);
uint8_t Button_read(uint8_t port_num, uint8_t pin_num);
void Button_with_interrupt_init(EN_EXT_INTERRUPT_t e, EN_INTERRUPT_TRIGGER_t t,
void (*ISR)());
void __vector_1(void) __attribute__ ((signal, used)); // ISR for external interrupt 0
```

MOTOR

```
void MOTOR_ALT_INIT(uint8_t IN1_port, uint8_t IN1_pin,
                    uint8_t IN2_port, uint8_t IN2_pin,
                    uint8_t IN3_port, uint8_t IN3_pin,
                    uint8_t IN4_port, uint8_t IN4_pin,
                    uint8_t ENA_port, uint8_t ENA_pin,
                    uint8_t ENB_port, uint8_t ENB_pin);
void MOTOR_ALT_FWD(float speed_ratio);
void MOTOR_ALT_BWD(float speed_ratio);
void MOTOR_ALT_LEFT(float speed_ratio);
void MOTOR_ALT_RIGHT(float speed_ratio);
void MOTOR_ALT_PULSE(float speed_ratio);
void MOTOR_ALT_STOP();
```