Software requirements specifications of car control system

* + Problem:
    - Controlling two motors of a car through four push-buttons using a microcontroller that has (DIO, Timer & PWM) modules.
  + Functional requirements:
    - our system has three speeds.
    - Button\_4 configures the speed and direction (forward and backward)
    - Pressing Button\_4 will change the car speed. Each button press must move to the next speed:
      * Speed 0: changes the car speed to 0%.
      * Speed 1: changes the car direction to forward, and speed to 30%.
      * Speed 2: changes the car direction to forward, and speed to 60%.
      * Speed 3: changes the car direction to forward, and speed to 60%.
      * Speed b: changes the car direction to backward, and speed to 30%.
    - Car won’t start move until speed and direction are configured first then only one of the other three buttons is pressed after the configuration:
      * pressing Button\_1 will make the car moves in the decided direction and speed as long as you press the button.
      * Pressing Button\_2 the car will turn right as long as you press the button.
      * Pressing Button\_3 the car will turn left as long as you press the button.
  + System modules:
    - Car control application module
    - Push button module
    - Motor module
    - DIO module
    - PWM module
    - Timer module
  + Modules interfaces (APIs):
    - DIO Module:
      * DIO\_ERROR\_t DIO\_PortInit (PORT\_t port, DIR\_t direction);
      * DIO\_ERROR\_t DIO\_PinInit (PORT\_t port, PIN\_t pin, DIR\_t direction);
      * DIO\_ERROR\_t DIO\_PinEnablePullUp (PORT\_t port, PIN\_t pin);
      * DIO\_ERROR\_t DIO\_SetPortVal (PORT\_t port, Byte value);
      * DIO\_ERROR\_t DIO\_SetPinVal (PORT\_t port, PIN\_t pin, Byte value);
      * Byte DIO\_GetPortVal (PORT\_t port);
      * Byte DIO\_GetPinVal (PORT\_t port, PIN\_t pin);
    - Timer Module:
      * TIM\_ERROR\_t TIM\_Init (TIM\_SELECT\_t Timer, CLK\_SELECT\_t clock, TIMER\_MODE\_t mode);
      * TIM\_ERROR\_t TIM\_SetTimerOVFVal (TIM\_SELECT\_t Timer, u8 u8TimVal);
      * TIM\_ERROR\_t TIM\_SetTimerOCVal (TIM\_SELECT\_t Timer, u8 u8TimVal);
      * TIM\_ERROR\_t TIM\_SetTimerOVFAction (TIM\_SELECT\_t Timer, void(\*callback)(void));
      * TIM\_ERROR\_t TIM\_SetTimerOCAction (TIM\_SELECT\_t Timer, void(\*callback)(void));
    - PWM Module:
      * PWM\_ERROR\_t PWM\_Init (TIM\_SELECT\_t Timer, PWM\_Mode\_t Mode);
      * PWM\_ERROR\_t PWM\_Start (duty\_cycle, frequency);
    - PushButton Module:
      * PSHBTTN\_ERROR\_t PSHBTTN\_Init (PORT\_t port, PIN\_t pin, PULLUP\_Status\_t status);
      * PSHBTTN\_ERROR\_t PSHBTTN\_EnablePullUp (PORT\_t port, PIN\_t pin);
      * Byte PSHBTTN\_Status (PORT\_t port, PIN\_t pin);
    - Motor Module:
      * MOTOR\_ERROR\_t Motor\_Init (PORT\_t port, PIN\_t pin);
      * MOTOR\_ERROR\_t Motor\_Start (PORT\_t port, PIN\_t pin, SpeedVal\_t speed, MDIR\_t direction);
      * MOTOR\_ERROR\_t Motor\_Stop (PORT\_t port, PIN\_t pin);
  + System layers (Layered architecture):
    - Application Layer
    - Electronic Unit Abstraction Layer
    - Microcontroller Abstraction Layer

|  |  |  |  |
| --- | --- | --- | --- |
| APP. | APPLICATION LAYER | | |
| ECUAL | PUSH BUTTON | MOTOR | |
| MCAL | DIO | PWM | TIMER |
| MICROCONTROLLER | | | |