ASU Racing Team’s

Formula Electric ICE – Embedded Development Task Documentation

Boula Nashat

Used Microcontrollers and Development Tools:

To have a good variation in coding style, I've used two micro-controllers to program with two different programming languages.

- Atmega16

  Ive used it because I have drivers ready to use with it as well as it being easy to simulate on proteus.

  Used Tools:

Eclipse Proton	2018-9	As an IDE to develop and debug the code
WinAVR	20100110	As a tool-chain to compile to and target Atmega15
Proteus	8.5	To simulate the execution of the application

- ESP32 with MicroPython Firmware

  I've used it because I already have the board.

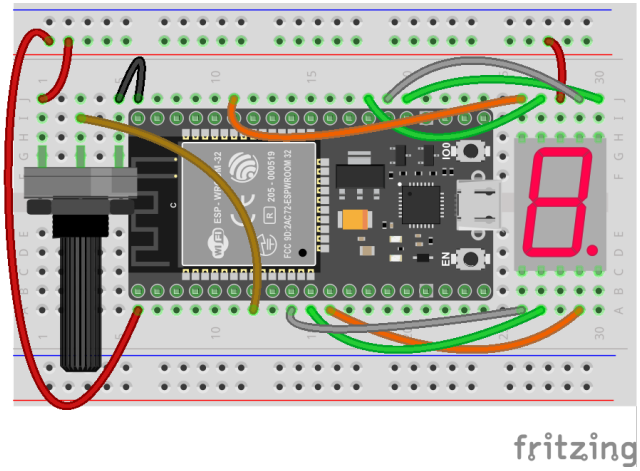
  Used Tools:

esptool	2.5.0	To change the board's firmware
rshell	0.0.15	To upload code files to the board
fritzing	0.9.3b	To draw a project schematic

Task One:

Write a code to interface with a seven segment. Get an ADC value from a potentiometer and map this value from0→5 then display it on the seven segment.

Configuration:



Hardware:

ESP32 – Seven-segment Display – 100kΩ Potentiometer

Code:

https://github.com/BoulaZa5/formula-electric-ice-embedded-development-task/blob/master/potentiometer-to-7-seg/potentiometer-to-7-seg.py

Video:

https://youtu.be/QepSwnirlLw

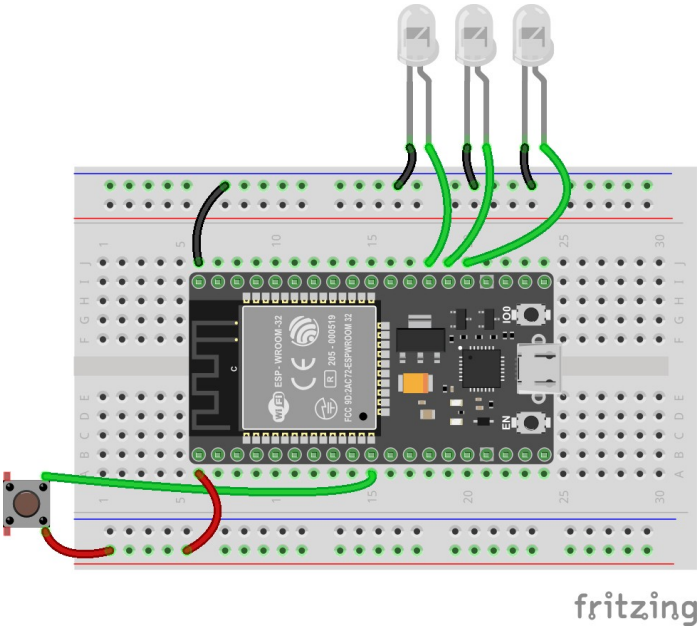
Task Two:

Build a project with 3 LEDs and a button, initially the 3 LEDs are off.

- ✔ *First button click time, LED1 on, LED2 off, LED3 off.*
- ✔ *Second button click LED1 blinking slowly, LED2 is on, LED3 off.*
- ✔ *Third button click LED1 blinking fast, LED2 blinking slowly, LED3 is on.*
- ✔ *Fourth button click LED1 off and LED2 blinking fast, and LED3 blinking slowly.*
- ✔ *Fifth button click LED1 off, LED2 off, LED3 blinking fast.*
- ✔ *Sixth button click resets to the initial state and start over again.*

You should use external interrupts and timers as needed.

Configuration:



Hardware:

ESP32 – 3 LEDs – Push Button

Code:

https://github.com/BoulaZa5/formula-electric-ice-embedded-development-task/blob/master/interrupts-to-leds/interrupts-to-leds.py

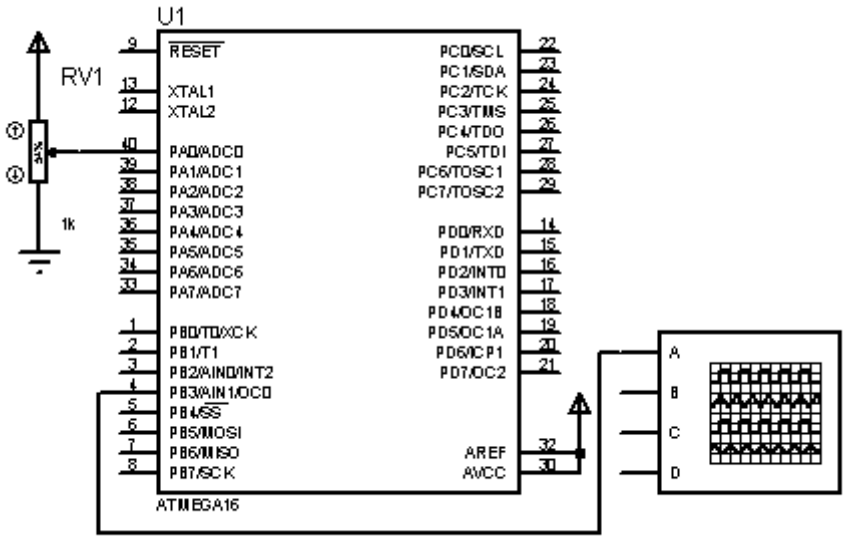
Video:

https://youtu.be/f2qTOg_trI4

Task Three:

Write a code for a microcontroller of your choice that initializes a GPIO pin to be output, reads an analog inputfrom ADC and initializes a timer to output a PWM signal with a duty cycle that changes from 0% to 100% basedon the ADC value (like a potentiometer changing the brightness of an LED with PWM).

Configuration:



Hardware:

Atmega16 – Oscilliscope – 1kΩ Potentiometer

Eclipse Project and Proteus Simulation files:
https://github.com/BoulaZa5/formula-electric-ice-embedded-development-task/tree/master/adc-to-pwm

Video:

https://youtu.be/qOxgV62MBdw

Task Four:

Write a code using PWM and overflow timers, external interrupts and GPIO configuration. Also using theinterface with an LCD and a keypad do the following:

Case 1:

The LCD will display the string "Enter your password", If the user entered the correct password a message will be displayed on the LCD "Correct", agreen LED will be turned on and a motor will be initiated.

Case 2:

The LCD will display the string "Enter your password", If the user entered a wrong password for less than three times a message will be displayed on the LCD "Wrong, Re-enter the password", After the 3 rd trial a red LED will blink, and a buzzer will buzz using an overflow times every1second

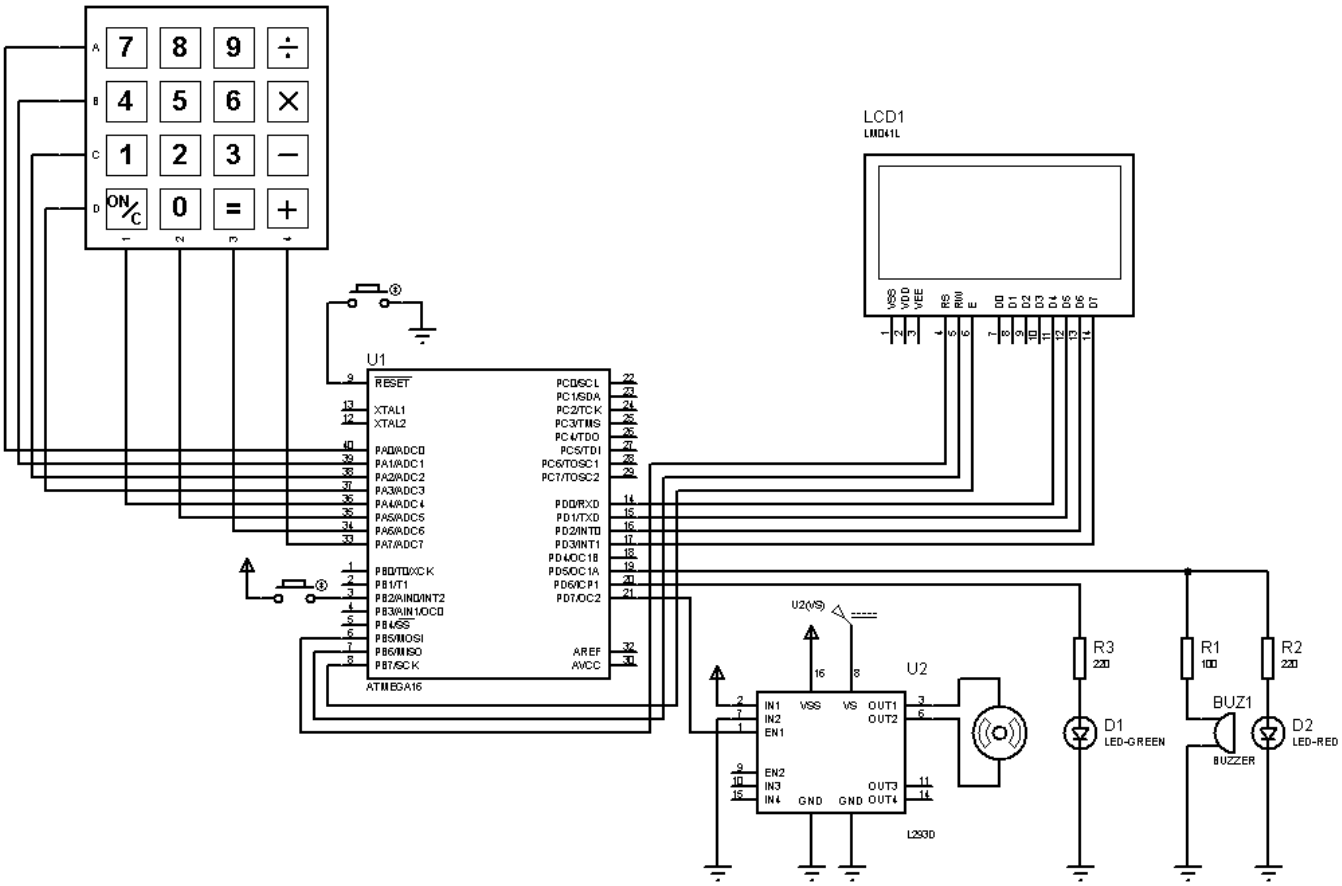
Note:

- Set the password: 123

- Using the delays is not permitted except for the interfacing with the LCD.

- Use an external interrupt to shut down the system using a push button.

Configuration:



Hardware:

Atmega16 – 2 Push Buttons – Small-Calculator Keypad – LM041L LCD – L293D Motor Driver – DC Motor – 100Ω Resistor – 2 220Ω Resistors – Buzzer – Red Led – Green Led

Eclipse Project and Proteus Simulation files:

https://github.com/BoulaZa5/formula-electric-ice-embedded-development-task/tree/master/keypad-to-lcd

Video:

https://youtu.be/XSd8cPhz5dc