# Lab 1

Kyle Jeffrey

**Abstract**—The first lab of the class introduces the PSoC Creator software for programming the PSoC 6 Microcontroller.

## 1 Introduction

This lab focuses on basic functionality of the PSoC 6 Microcontroller. This includes the initialization of GPIO pins, Counters, Timers, and registers as well as building familiarity with the API of the on board systems. The final section of the lab builds a proximity sensor using the HC-SR04 ultrasonic sensor. This part emphasized configuring hardware components of the microcontroller to handle the repitive tasks that could waste cpu cycles.

## 2 Part 1: GPIO Pin Toggling

The PSoC 6 Microcontroller has many GPIO Pins that can be configure as Digital/Analog Input/Output. This section investigates toggling a GPIO pin digital out using a direct write or using a control register.

### 2.1 Direct Write

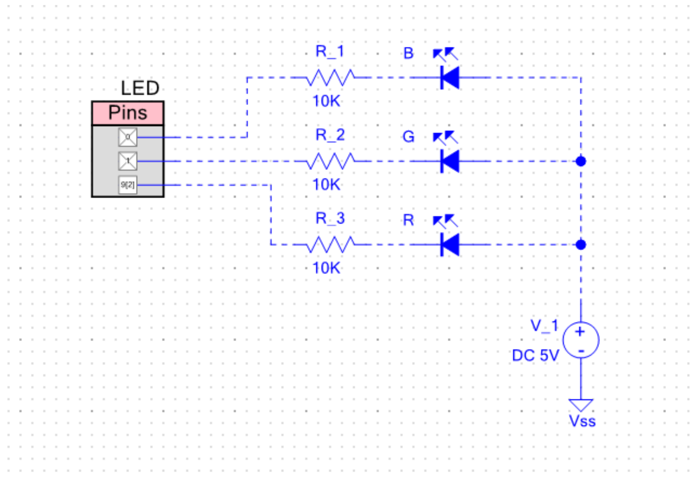


Fig. 1. Hardware Setup

The design uses a 4 pronged LED provided from the lab kit that has 3 channels and an input for Red, Green, and Blue LEDs. As shown, the LED's input's are tied to the Vdd voltage supply from the Microcontroller at 3.3V, with a resistor at their outputs to limit the current according to datasheet specifications. 10K resitors were used here. Each LED is then tied to a GPIO input. The main loop uses a 50 ms delay to make the shifting colors visible. Remembering that setting a GPIO pin to logic high turns the LED off, the LED's have 8 different color permutations from Red, Green, Blue combinations.

The design was pretty straightforward so not much testing was needed. It began by using one LED using a direct GPIO write and then adding the rest.

### 2.2 Register Write

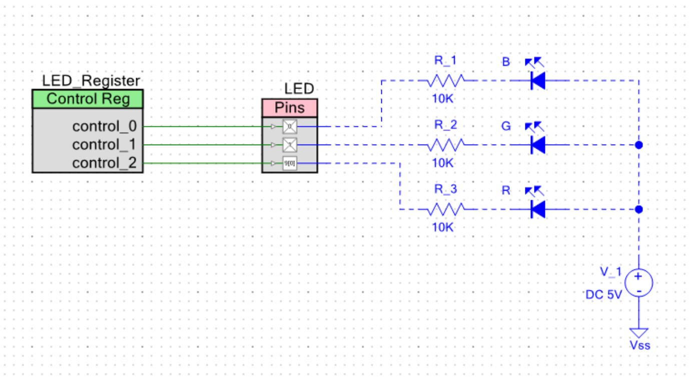


Fig. 2. Hardware Setup

Instead of direct writing to values to the digital out GPIO pins, this section used a hardware input for the Pins and a control register. The control register's value were then set in code with a single line for writing a 3-bit value, vs individual setting the value's of each GPIO digital output.

## 2.3 Waveforms

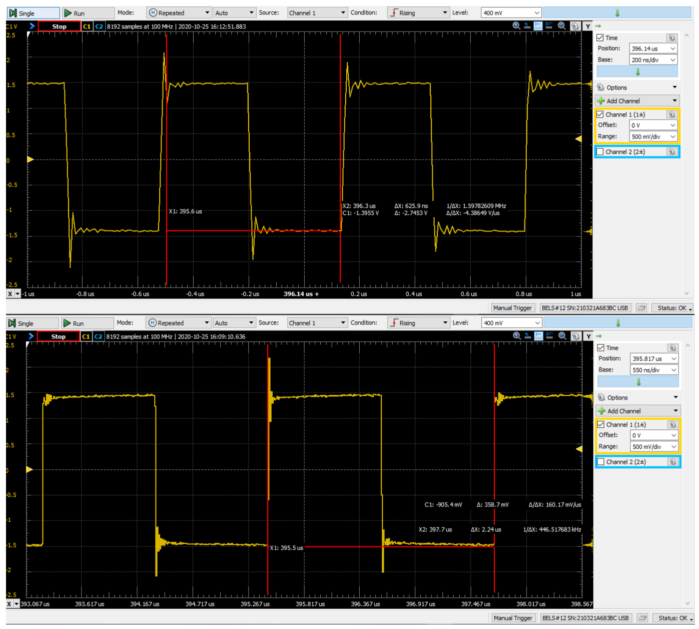


Fig. 3. Write Speed, Top = Direct Write, Bottom = Register Write

Here is a waveform of the GPIO Pin outputs, showing the difference in output speed. Notice the speed increase of direct write to the GPIO Pin's versus writing to the control register. This could be due to writing a full byte to the Register. The logic to write a full byte could be more complicated than writing direct 1's or 0's to the GPIO.

## 3 LED Brightness Using PWM

This section uses a potentiometer to control the brightness of an LED. The potentiometer reading is used converted by an analog to digital convereter(ADC), and the value was then used to modulate a PWM signal.

The ADC continuously reads the value of the potentiometer in the main loop and uses that value to change the duty cycle of the ADC. In the configuration for PSoC this is done by changing the compare value and period of the PWM component. The duty cycle then is $\frac{comparevalue}{period}$

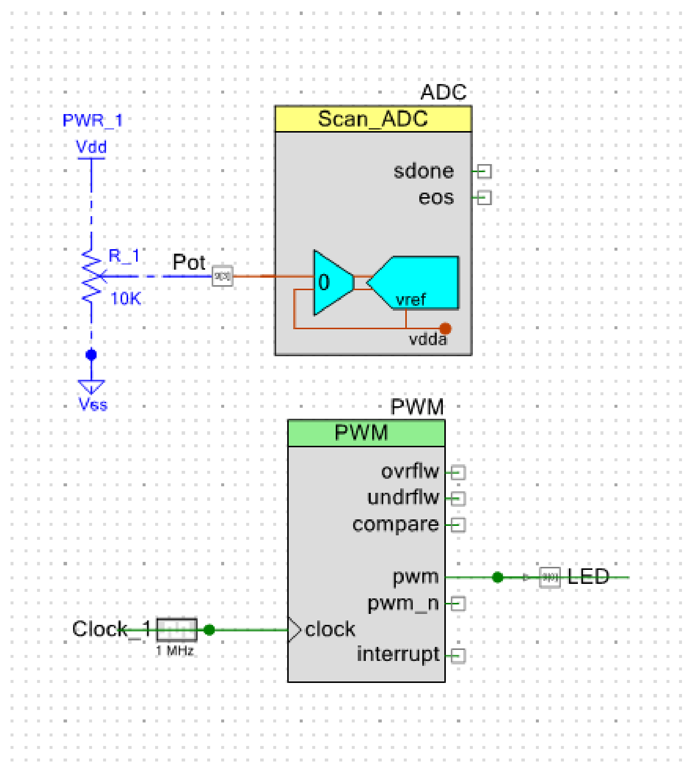


Fig. 4. Schematic

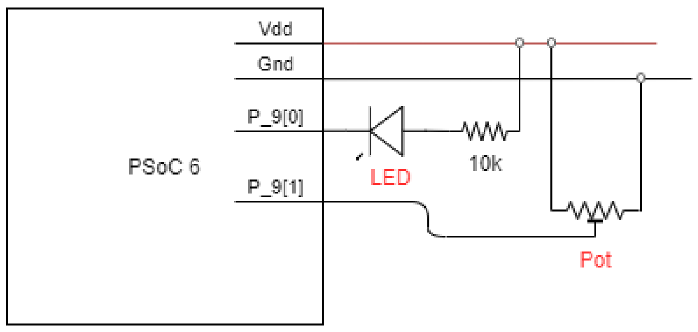


Fig. 5. Top Level Design

## 4 Proximity Detector

This section uses the HC-SR04 Proximity sensor. The sensor measures distance from an object with the equation $Distance(cm) = PulseWidth/58$. The sensor is powered by 5v and requires a 10uS pulse to trigger ultrasonic waves that are then read from the echo pin of the sensor. Figure (6) shows the datasheet timing diagram of the sensor trigger and echo response.

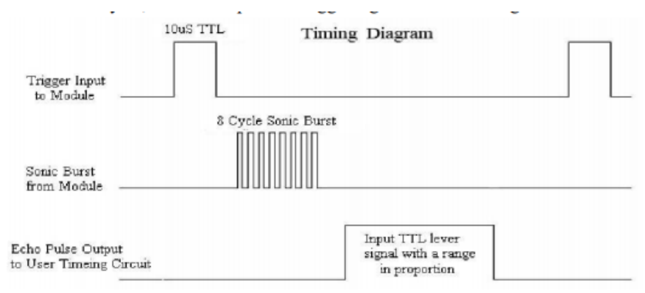

10uS TTL
Timing Diagram
Trigger Input to Module
8 Cycle Sonic Burst
Sonic Burst from Module
Echo Pulse Output to User Timing Circuit
Input TTL lever signal with a range in proportion

Fig. 6. HC-SR04 Sensor Trigger and Response

## 4.1 Design

Notice in the schematic the two primary components *the Trigger Counter and the Echo Counter*. The trigger counter is what periodically sends a 10uS pulse to the Ultrasonic sensor and the Echo Counter measures the width of the response pulse. The trigger signal is sent every 500 ms by setting the Trigger Register to high then setting the Reload Register controlling the Trigger Counter to high and back to low. The Trigger Counter is configured to send a pulse when it counts to ten, which is 10uS based on its clock configuration. This is how a 10uS pulse is achieved. The Echo Counter is then used with an interrupt to measure the echo pulse width.

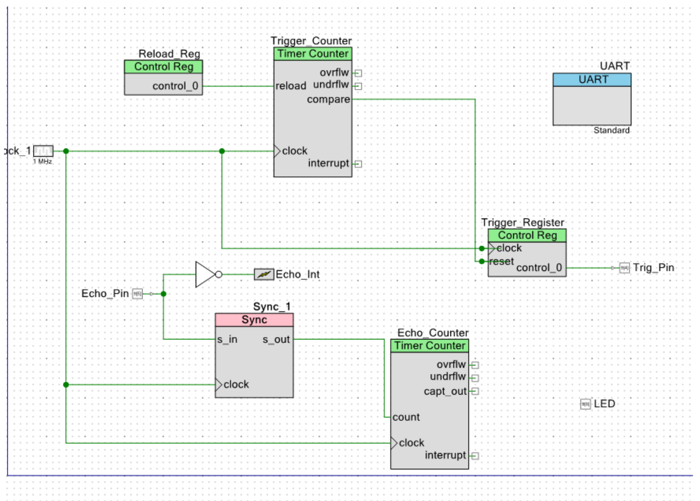


Fig. 7. Schematic

Lastly, because the Sensor uses a 5v source, all of the signals sent to the sensor need to be 5v and the signals recieved will be up the same, so a voltage level converter was used to step up and step down the voltage accordingly.

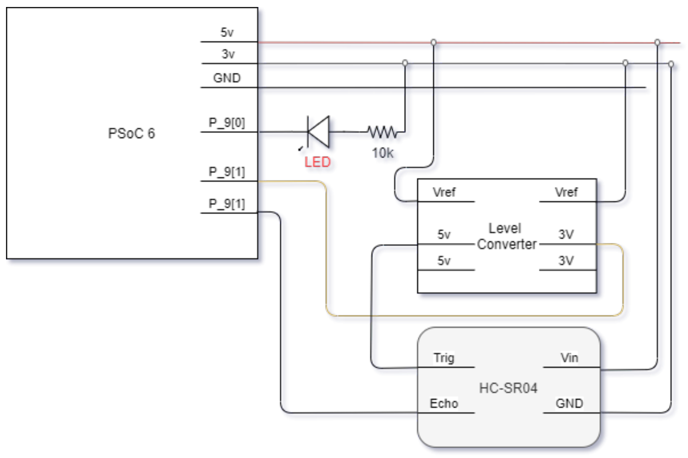
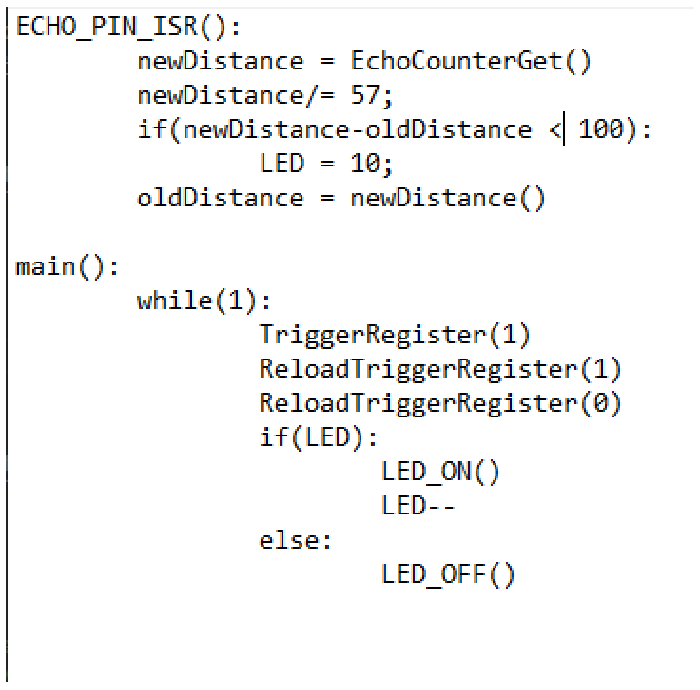


Fig. 8. Top Level Design

## 4.2 PseudoCode

If an object is detected within 100 cm, the LED will turn on and stay on for 10 cpu cycles, or 5 seconds.



```
ECHO_PIN_ISR():
        newDistance = EchoCounterGet()
        newDistance/= 57;
        if(newDistance-oldDistance < 100):
                LED = 10;
        oldDistance = newDistance()

main():
        while(1):
                TriggerRegister(1)
                ReloadTriggerRegister(1)
                ReloadTriggerRegister(0)
                if(LED):
                        LED_ON()
                        LED--
                else:
                        LED_OFF()
```

Fig. 9. PseudoCode

## 5 Conclusion

This lab was an introduction into the PSoC 6 production environment, with a minor introduction into sensors. Part 1 might not have any practical use but Parts 2 and 3 could. For part 2, learning how to modulate pwm signal's using a potentiometer input could be used for any measure of applications, like a DC motor or an LED with modulated brightness. For part 3, ractically, a proximity sensor with a half second polling rate could run at very low power and left on. It could be used to determine whether to run a security camera or maybe to unlock a door.

**Kyle Jeffrey** is a Senior Robotics Engineering Student at the University of California Santa Cruz. He is the Secretary of the Engineering Fraternity Tau Beta Pi and the lead Hardware Engineer at the on campus startup Yektasonics.