EGR: 226 Microcontroller Programming and Applications

Winter 2021

Instructor Prof. Trevor Ekin

**Lab 11: Precise Timing Intervals using Infrared (IR) Signals**

Gabriel Gasbarre

April 14th, 2021

Contents

[1. Objectives 3](#_Toc69310782)

[2. Equipment 3](#_Toc69310783)

[3. Introduction 4](#_Toc69310784)

[3.1 Pre-Lab 4](#_Toc69310785)

[3.3 Part 2 - Building the detector circuit 5](#_Toc69310786)

[4. Procedure: 5](#_Toc69310787)

[4.1: Part 1 - Building and Detecting the IR 5](#_Toc69310788)

[4.2: Part 2- Building the detector circuit 6](#_Toc69310789)

[5. Results: 8](#_Toc69310790)

[5.1 Part 1 Results 8](#_Toc69310791)

[5.2: Part 2 Results 8](#_Toc69310792)

[6. Conclusions. 8](#_Toc69310793)

1. Objectives

Learn how to generate an IR signal;

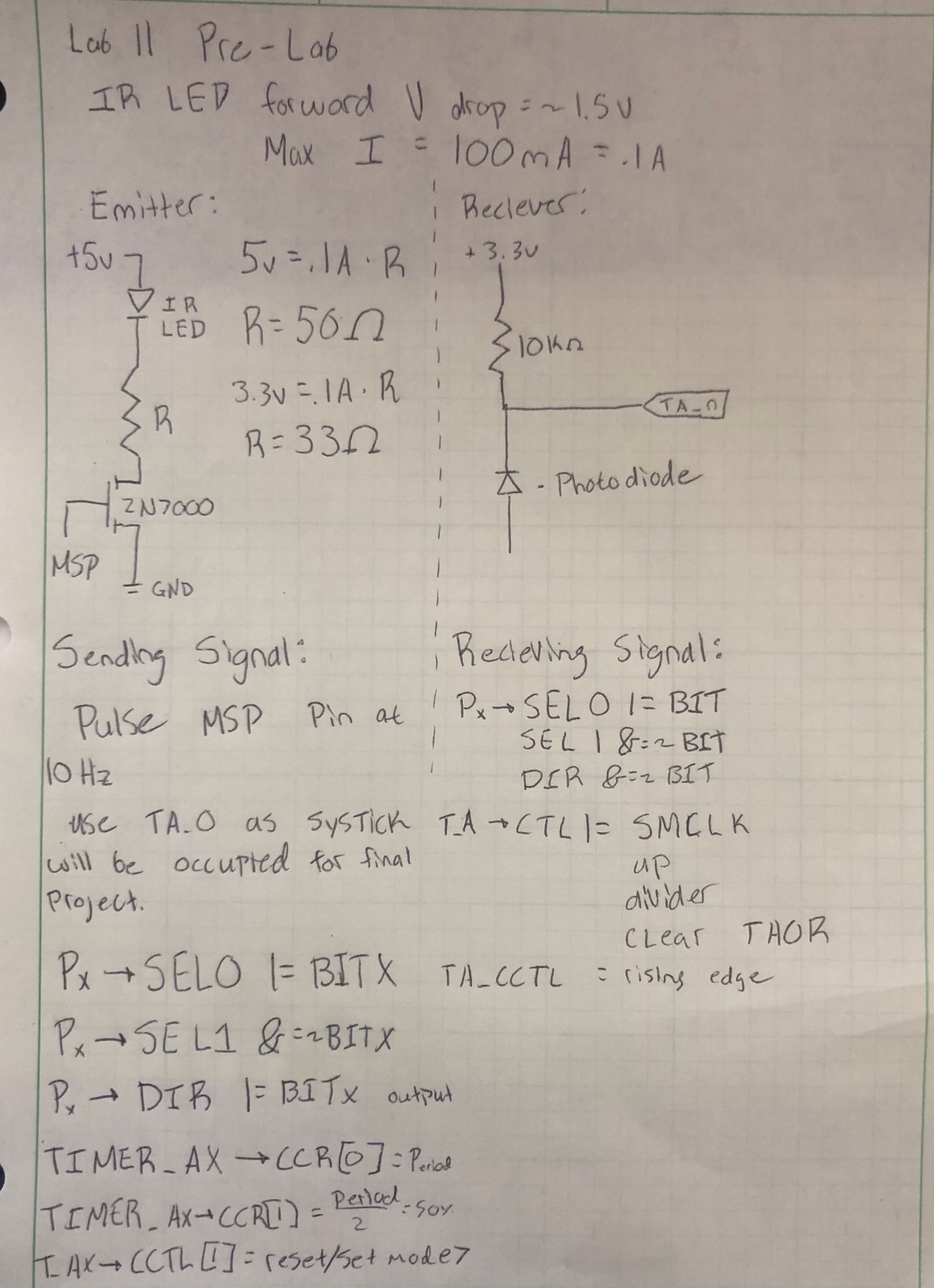
Learn how to detect an IR signal.

1. Equipment

|  |  |  |
| --- | --- | --- |
| **Part** | **Description** | **Model** |
| CCS (Code Composer Studio) | Integrated development environment to develop applications for Texas Instruments embedded processors. | 10.0.00010 |
| MSP432 | Mixed-signal microcontroller family from Texas Instruments. | MSP432P401x |
| EGR:226 Lab 6 Exercise | Interfacing a keypad with the MSP432 | N/A |
| Photodiode Infrared LED’s | electronic components that emit and receive infrared light | N/A |
| MOSFET Transistor | Insulated-gate field-effect transistor | 2N7000 |

1. Introduction

## 3.1 Pre-Lab



## 

3.2: Part 1- Building and Detecting the IR

For part 1 of the lab, students are to use their IR LED circuit created in the pre lab in order to create the hardware setup for the emitter portion of the lab. This emitter should be controlled by a GPIO pin using a MOSFET. This GPIO pin should be configured using a timer (TimerA) in order to send a 10Hz signal that will eventually be received by the IR Receiver. Because IR wavelengths are unable to be observed with the human eye, it is suggested at this point to either hook up an LED in the visible light spectrum or look at the IR LED with a camera in order to ensure its successful operation.

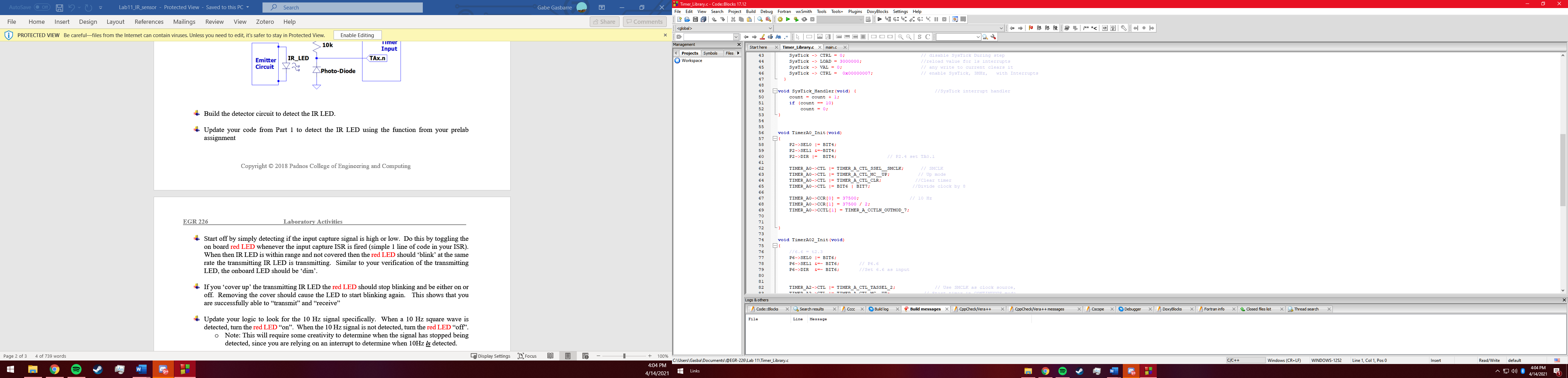
## 3.3 Part 2 - Building the detector circuit

For part 2 of the lab, students are to hook up the IR receiver circuit created in the pre lab to initially detect a high or low input signal. This is confirmed by turning on the on-board LED whenever a signal is successfully received. This program / circuit should then be expanded to look for the 10Hz signal sent by the emitter specifically. This means that, when the IR receiver either is not receiving a signal, or if the signal is not 10Hz, the on board LED should be off. If, however the correct signal is being received, the LED should remain on.

# 4. Procedure:

4.1: Part 1 - Building and Detecting the IR

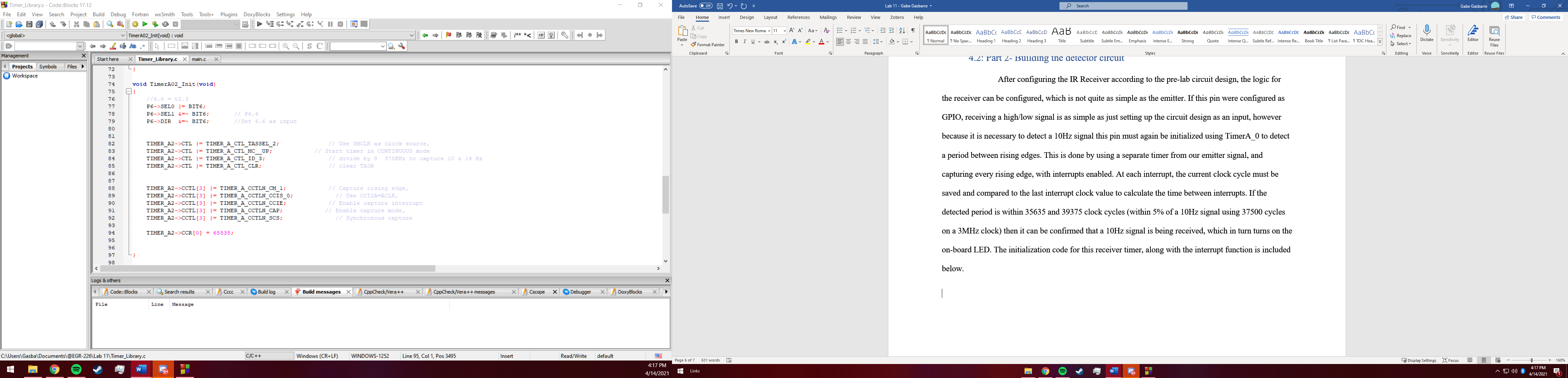
After creating the circuit described in the pre-lab, the logic for the IR Emitter must be configured using some sort of timer in order to send a 10Hz signal. For this specific application, the TimerA\_0 will be used as it is hooked up to a specific pin and is moderately reliable. The initialization for this is very straightforward and has been completed many times before in previous labs. Included below is the code that automatically activates the IR Emitter at 10Hz intervals.



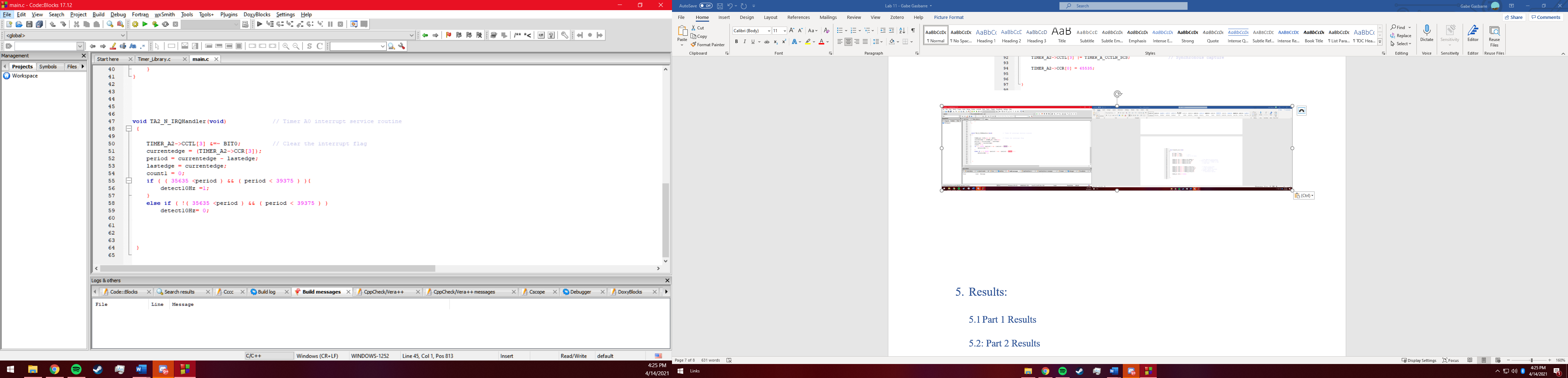
4.2: Part 2- Building the detector circuit

After configuring the IR Receiver according to the pre-lab circuit design, the logic for the receiver can be configured, which is not quite as simple as the emitter. If this pin were configured as GPIO, receiving a high/low signal is as simple as just setting up the circuit design as an input, however because it is necessary to detect a 10Hz signal this pin must again be initialized using TimerA\_0 to detect a period between rising edges. This is done by using a separate timer from our emitter signal, and capturing every rising edge, with interrupts enabled. At each interrupt, the current clock cycle must be saved and compared to the last interrupt clock value to calculate the time between interrupts. If the detected period is within 35635 and 39375 clock cycles (within 5% of a 10Hz signal using 37500 cycles on a 3MHz clock) then it can be confirmed that a 10Hz signal is being received, which in turn turns on the on-board LED. The initialization code for this receiver timer, along with the interrupt function is included below.

Initialization:



Interrupt:



# Results:

* 1. Part 1 Results

Part one of the lab was quite straightforward as setting up an LED on a timer is something that has been completed in part or in full multiple times throughout all of the labs. The only real issue with this pat is confirming that, in fact, your IR LED works as it Is impossible to see IR light with the human eye. Aside from this however there were really no issues setting up this portion of the lab.

5.2: Part 2 Results

Part two of the lab was arguably the most difficult lab portion to date for a multitude of reasons. First, initializing a TimerA that had not yes been used before was confusing as configuring the initialization had many different variables to change, such as capturing a rising edge, synchronous capture, and using different CCTL values. Next came the issue of detecting the period of the signal being received. In order to see what the period of the received signal is in real time; it is necessary to run a print statement. This, however, slows the Timer\_A clock speed and oftentimes ruins any sort of value that was going to be received. Next, As the initial goal was to use libraries for a more organized code, many of the calculation variables were being stored as global variables that turn out to be relatively unreliable as their update times often ruin whatever inputs were desired to be calculated. After all of these issues were resolved, eventually a pseudo-working program was configured that is probably about 90% reliable, with some optimization still needed.

# Conclusions.

This was certainly what could be described as an entire waste of time in many aspects, however it did create a much better understanding of not IR receivers/emitters, but instead the inner workings of the MSP432 tiers and variable storage that will ultimately assist with precise timing input in the future.