**Wellington Institute of Technology**

**School of Engineering**

**MG7013 Embedded Systems**

**Project 1**

**Flicker Fusion Threshold**

**Student: Joven Sepulveda**

**Due Date: 30 April 2019**

**Tutor : Frank Beinersdorf**

**Table of Contents**

1. Introduction. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3
2. Method
   1. Design process for circuit. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 4
   2. Design process for program . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5-7
3. Result
   1. Oscilloscope output. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7
   2. Serial monitor output. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
4. Conclusion
   1. Limitations of project . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 8
   2. Possible future improvements. . . . . . . . . . . . . . . . . . . . . . . . . . 8

Appendices

1. **Introduction**

The project is to design an embedded system use to measure the blinking light frequency, particularly the flicker fusion threshold which human perception of blinking light to be steady. The design involves the prototyping of circuit and programming Teensy microcontroller with Visual Studio Code application in C++ language. Aside from microcontroller, the circuit components utilize LED as blinking light, potentiometer as adjustment of blinking frequency and push button switch as an interrupt to capture the blinking frequency and display to serial monitor of computer. Moreover, the project aims to design a program to control the blinking frequency adjustable with digital setting and to demonstrate method how to use external interrupt and debounce to filter the unwanted interrupt or noise.

1. **Method**
2. **Design process for circuit**

Select three pins of teensy microcontroller for the connection of components. The design needed digital output to LED, analog input from potentiometer, and digital input from push button switch. Set up these components onto the breadboard. Look up datasheet for LED to know its specification and operation. Then, wire the LED anode to ground and cathode to two parallel resistors, each rated 360Ω. This parallel resistor with total resistance of 180Ω connected the other end to digital output pin mention previously. This resistance needed for forward current 10mA of the LED. By ohms law, this limiting current assume from resistance 130Ω and resistors voltage of 1.3V. These resistors voltage calculated from voltage of pin mention above at 3.3V and minus with LED typical voltage 2V. Next, the potentiometer connected from wiper pin to microcontroller analog input. This input has an ADC (analog to digital converter) from voltage to binary number. The other two pins of potentiometer connected to 3.3V and ground. Finally, the push button switch connected to digital input pin and ground of microcontroller. The circuit diagram is shown below.

**Circuit Diagram**

1. **Design process for program**

Define the pin constant and variable for values at general declaration. Next, setup the main function for the serial monitor data rate, pin modes and analog resolution. Then, design two functions each for LED blink and for push button external interrupt and debounce. LED blink function have 50% duty cycle with duration taken from value of ADC in microsecond. The LED state from LOW to HIGH and from HIGH to LOW will alternately change with this duration. Since analog resolution is set to 16 bits, ADC value will be from 0 to 65,536. This value is considered half of the duration or period; therefore, frequency can be calculated within this function and stored to global variable called “frequencyValue”. Check the operation of this function by adjusting the potentiometer and observe the LED flicker. The second function, pushbutton, is call as interrupt service routine (ISR) for attach interrupt function. The pushbutton function need digital read of the state of the button to compare the previous state. If not the same state timer for the debounce delay reset. This method removes unwanted signal cause by switch make and break of contact. Check the operation of this function by pressing the button and ensure no unintended serial monitor output. Next, back at main setup function, invoke the attach interrupt function include pushbutton function as its ISR. Finally, the main loop function, include the constant analog read for potentiometer pin to get ADC value and LED blink function call.

**Source Code**

#include <Arduino.h>

/\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Project 1: Flicker Fusion Threshold \*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

This is an embedded system project which measure flicker fusion threshold.

The measure of frequency on how much human perception of LED blinking.

The circuit:

1. Teensy 3.1/3.2 Microcontroller

2. 2pcs. 360Ω resistor in parallel connected from Pin 12 to LED

3. this LED then connected to ground

4. 10kΩ potentiometer with wiper connected from Pin 14 or A0

and other two terminal from 3.3V to ground

5. a push button switch connected from Pin 11 to ground

Created: 19 April 2019 by Joven Sepulveda

Weltec School of Engineering

MG7013 Embedded Systems

\*/

// constant set pin numbers:

const uint8\_t LEDPIN = 12; // output pin for the LED

const uint8\_t POTENTIOMETERPIN = A0; // input pin for the potentiometer

const uint8\_t BUTTONPIN = 11; // input pin for the push button

// variables of changing values

uint16\_t adcValue = 0; // variable storage for ADC value from potentiometer

float frequencyValue = 0; // variable storage for calculated value of frequency

// forward declaration of function

void pushbutton(); // function for push button

void blinkLED(); // function for LED blinking

// initialize serial monitor, analog resoution, pin modes and interrupt function

void setup() {

Serial.begin(500000); // sets serial data rate to 500,000 bps

analogReadResolution(16); // sets resolution 16 bits, so analogRead() be 0 to 65536

pinMode(LEDPIN, OUTPUT); // declare the LEDPIN as an OUTPUT

pinMode(BUTTONPIN, INPUT\_PULLUP); // declare the BUTTONPIN as an INPUT PULLUP

attachInterrupt(digitalPinToInterrupt(BUTTONPIN), pushbutton, FALLING); // push button external interrupts

}

// loop checks analog value and will send output signal for LED blink

void loop() {

adcValue = analogRead(POTENTIOMETERPIN); // read the adcvalue from the potentiometer:

blinkLED(); // call function blinkLED

}

// 50% duty cycle LED blink and calculate frequency

void blinkLED(){

static uint32\_t previousTime\_micro = 0; // first call 0 for initial value for previousTime\_micro

if ((micros() - previousTime\_micro) >= (adcValue)){ // if elapse time is equal or greater than adcValue

// LED change state

if (digitalRead(LEDPIN) == LOW){ // if LED is off

digitalWrite(LEDPIN, HIGH); // switch LED on

} else { // else if LED on

digitalWrite(LEDPIN, LOW); // switch LED off

}

frequencyValue = (float) 1000000/((micros() - previousTime\_micro)\*2); // calculate frequency of LED blink

previousTime\_micro = micros(); // reset previousTime\_micro

}

}

// function for attachinterrupt at instant push button and allow debounce delay

void pushbutton(){

noInterrupts(); // disables external interrupts

const uint32\_t DEBOUNCEDELAY = 100; // constant debounce delay of 100ms

static uint32\_t previousTime\_ms = 0; // first call variable

static bool lastbuttonstate = LOW; // previous state from push button

// If state is due to noise or press

if (digitalRead(BUTTONPIN) != lastbuttonstate){

previousTime\_ms = millis(); // reset the previous time

}

// if elapse time is equal or greater than DEBOUNCEDELAY

if((millis() - previousTime\_ms) >= DEBOUNCEDELAY){

// serial monitor output

Serial.print("ADC Value: "); // print label

Serial.println(adcValue); // print ADC input value cause by potentiometer value

Serial.print("Frequency: "); // print label

Serial.print(frequencyValue, 2); // print frequency

Serial.println(" Hz"); // print label

}

lastbuttonstate = digitalRead(BUTTONPIN); // save the reading latest button state

interrupts(); // allow external interrupts

}

1. **Results**
2. **Oscilloscope**

Oscilloscope probes connected to digital output to resistor and LED and its ground. This will measure the frequency of LED’s blink. My flicker fusion threshold is shown below display.

1. **Serial Monitor Output**

Serial monitor output captures the frequency reading as the button press and display on the computer screen. My flicker fusion threshold is shown result below almost same value from above result of oscilloscope.

1. **Conclusion**
2. **Limitations of the project**

This program works well for one button. However, for embedded system employed two or more digital input may need additional push button function. Using attach interrupt function for push button cannot guarantee reuse for another push button. This may create problem that affect the state of all push button using this attach interrupt function.

1. **Possible Future Improvement**

To improve against above limitation, the project can be revise in two ways. First, in general declaration using the constant push button pin into variables. For this reason it could change the push button pin. Another, by making the function as object oriented so the function can be reuse. Modify the function to suitable class module and convert it into a library. The later is the best solution for the problem above.