MEAM 510 Lab 2

2.1 Switches and Debouncing and Input Capture

2.1.1 Get a SPST switch from the mini store. Solder wires to it(solid core). Plug the wires from the switch along with a resistor into a breadboard (as in lecture) so that the state of the switch can be read by an input port on the teensy as either 0V or 5V. Write code that will read the state of the switch and make an LED turn on when the switch is depressed and turn off when the switch is not press. **Submit your code and a schematic of your switch, resistors, Teensy ports and LED.**

From the description of the question, to make an LED turn on when the switch is depressed and turn off when the switch is not press, a normally open SPST should be applied in the circuit, so that the code could be written in a way that is much easy to read.

In GM lab, this kind of SPST is Normally Open.

Fig 1 Normally Open SPST

I choose 160-1610-ND 5mm blue LED as my output. To ensure both the teensy and LED work safely, an appropriate resistor should be connected in series with LED. Forward Voltage vs Forward Current graph is shown as Fig.2.

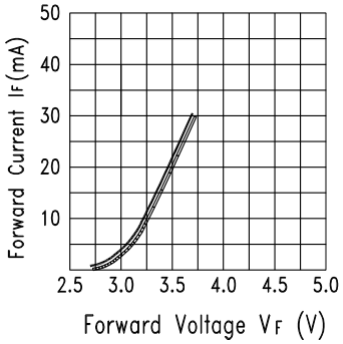

Fig 2 Forward Current vs Forward Voltage

Current flow through teensy is at most 20mA. In this case, the forward voltage of LED is 3.5V. Therefore, the LED should at most connect a resistor which is more than :

$$R_1 = \frac{5V - 3.5V}{20mA} = 75\Omega$$

Therefore, I connect a 150Ω resistor in series with the LED.

Also, the resistor connected to the input port should also be more than

$$R_2 = \frac{5V}{20mA} = 250\Omega$$

From teensy 2.0 datasheet, we know that there are two channels for timer input capture, which is D4 and C7. Choose C7 as the input port. By changing the register PINC we could configured port as input. The state of the port could be read using `bit_is_set(PINx, BIT)` or `bit_is_clear(PINx, BIT)`. Set PC7 as input and PD1 as output. The circuit diagram could be drawn as below

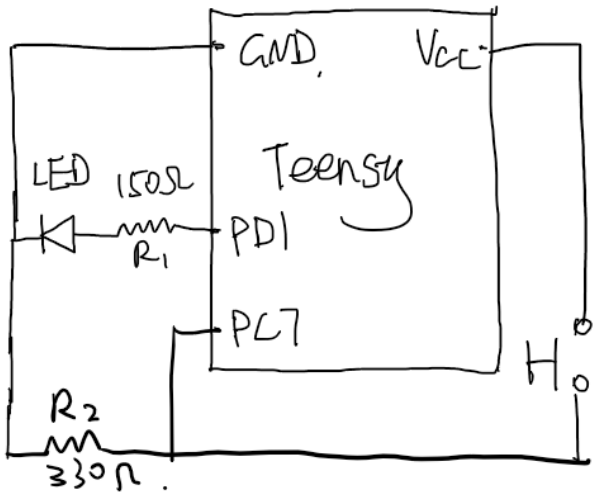

Fig 3 Circuit Diagram for 2.1.1

And the code is shown as below:

```
/* Name: main.c
 * Author: Yuchen Sun
 * Copyright: <insert your copyright message here>
 * License: <insert your license reference here>
 */

#include "teensy_general.h"  // includes the resources included in the teensy_general.h file

int main(void)
{
    set(DDRD, 1); // set PD1 as output
    clear(DDRC, 7); // clear value in PC7
    clear(PORTD, 1); // Take the output PD1 low
    /* insert teensy initialization here*/

    while(1)  // the main loop
    {
```

```
        if(bit_is_set(PINC, 7))
        {
            set(PORTD,1); // if there is input in PC7, set the output pin high
        }
        else
        {
            clear(PORTD,1);// if there is no input in PC7, clear the output pin low
        }
    }
    return 0; /* never reach*/

}
```

(Also in 2_1_1 folder)

2.1.2 Add to the code from 2.1.2 so that it prints to a terminal using the m_usb serial commands each time the switch is depressed and each time it is released. Turn on timer 3 setting the prescaler to divide by 1024 and print out a time stamp each time the switch changes state. Notice that when the switch is bouncing with each press you may get multiple printout with short time stamp values between them.

Add a resistor and capacitor low filter between the switch and the Teensy input port so that switch is "debounced". Make sure you have a large enough RC time constant so that you don't see multiple bounces on a single press, but not too large that you distort button presser that occur at roughly 10 times per second. To verify the effect of the RC circuit, use the output of a square wave from a function generator into the RC circuit and view the output. Notice how the signal changes as the frequency goes from below 10Hz to above 100Hz. **Show your calculation for cutoff frequency of your low pass filter.**

Use the input capture function of the timer on the Atmega32U4 to measure how fast you can depress a switch. Change the code so that it prompts the person when to start, and then measures 5 presses and prints out the average time between the 5 presses in milliseconds.

Submit your fastest time, your code and a schematic of your switch, resistors, capacitor, Teensy ports.

Code for this problem is in the folder 2_1_2.

When there is no capacitor in the circuit, the frequency of bouncing is 357Hz, measured by the oscilloscope, which means that in the debouncing frequency of low pass filter should be less than 357Hz and more than 10Hz

Choosing a low pass filter with a cut off frequency of 50 Hz.

A low-pass circuit is shown in Fig 4. When the SPST is open, the input pin could not measure anything. When the SPST is closed, there should measure the voltage of Vcc.

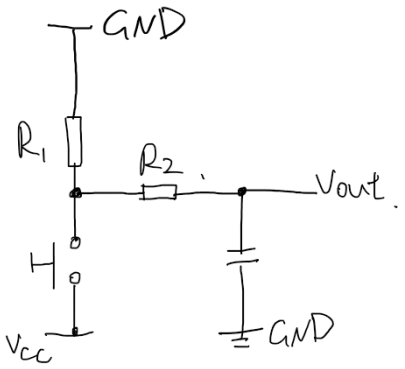

Fig 4 Low Pass Filter

Connect the low pass filter to teensy, and set PC7 as the input port. The circuit diagram could be draw as below,

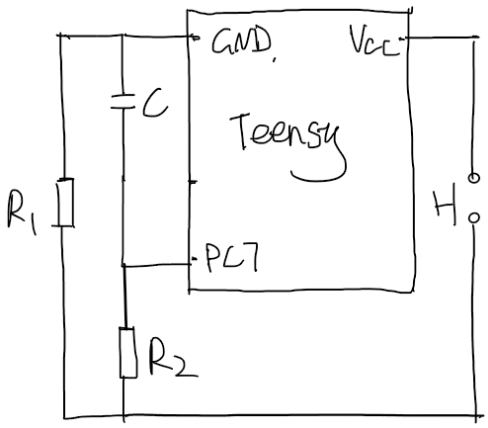

Fig 5 Circuit Diagram for 2.1.2

In the above diagram, set $R_1 = 330\Omega$ and $R_2 = 330\Omega$. We know that the cut-off frequency could be calculated as below:

$$f = \frac{1}{2\pi R_2 C}$$

Set the cut-off frequency to 50HZ, we have: $C = \frac{1}{2\pi R_2 f} = 9.64 \times 10^{-6}$. Thus, choose a $10\mu F$ capacitor. When $R_2 = 330\Omega$ and $C = 10\mu F$, the cut-off frequency is:

$$f = \frac{1}{2\pi R_2 C} = \frac{1}{2 \times 3.14 \times 330 \times 10 \times 10^{-6}} = 48.2Hz$$

From the calculation, we could easily know that only frequency below 48.2Hz could be detected by the low pass frequency.

Using the code below to test the function of low pass filter. I found out that when the frequency of a square wave is lower than 48 Hz, the terminal could print out the time between each hit correctly. As the frequency goes up, the terminal prints very fast, but the time is still valid. However, when the frequency reach 240Hz, the terminal slows down, and when the frequency is 250Hz, the terminal

could not print anything. I think it is because in the low pass filter circuit, the high frequencies are attenuated as the frequencies goes higher and higher. Therefore, the teensy could still print values when the frequency is over 50Hz. However, when the frequency is near 250Hz, the amplitude of the square wave is attenuated to 0 and could not be read by teensy.

My fastest time is 110ms, as it is shown in Fig 6.

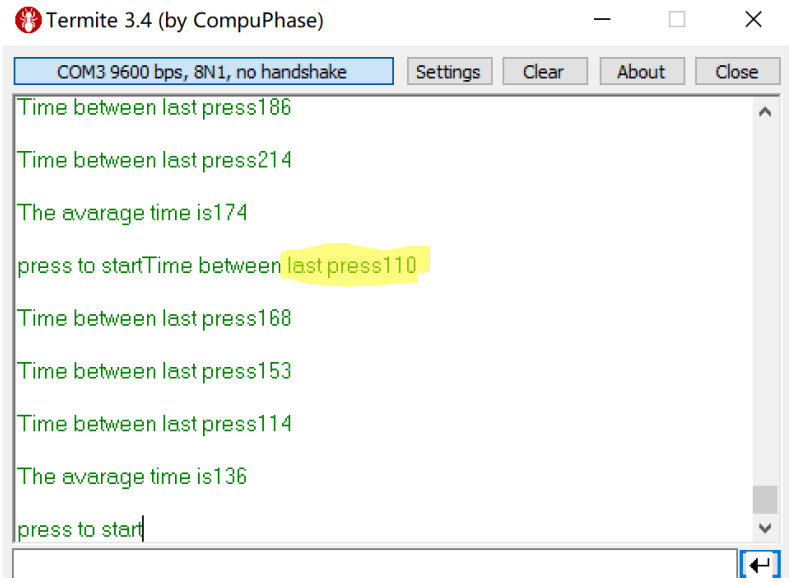

Fig 6 Fastest Time of Pressing

Extra credit:

The finest time resolution multiplies maximum counts should be larger than the slowest time. Teensy could at most counts $2^{16} = 65536$ times. So the finest time resolution is:

$$f = \frac{65535}{0.5} = 131070Hz$$

$$\frac{16\text{M}}{131070} = 122$$

, which means that the prescaler should be greater than 122. Select the \256 prescaler. The time resolution is:

$$\frac{16\text{M}}{256} = 62.5KHz$$

The code is in 2_1_2extra folder.

2.2 Phototransistors

2.2.1

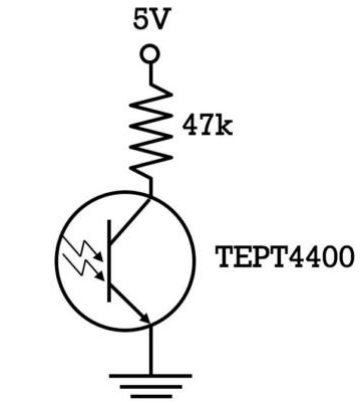

Fig 7 circuit diagram for TEPT4400

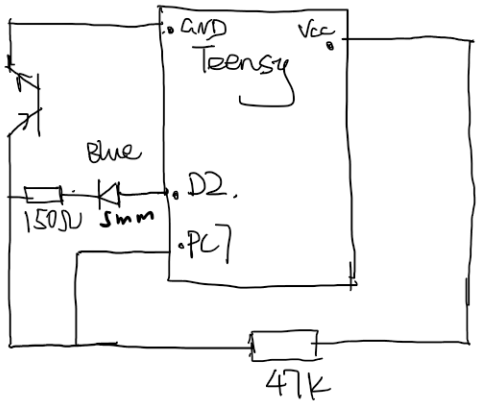

Fig 8 Circuit Diagram for Problem 2.2.1

First measure the output of the sensing circuit and change the resistor value into something 10x larger and 10x smaller. The output voltage is shown as below:

Table. 1 Output Voltage of Sensing circuit

Resistors(Ω)	47K	470K	4.7K
Light covered(V)	4.3	3.7	4.3
Light uncovered(V)	3.9	2.0	2.3
Flashlight(V)	0.298	0.29	0.30

In teensy, the logical high/low is defined by the following diagram.

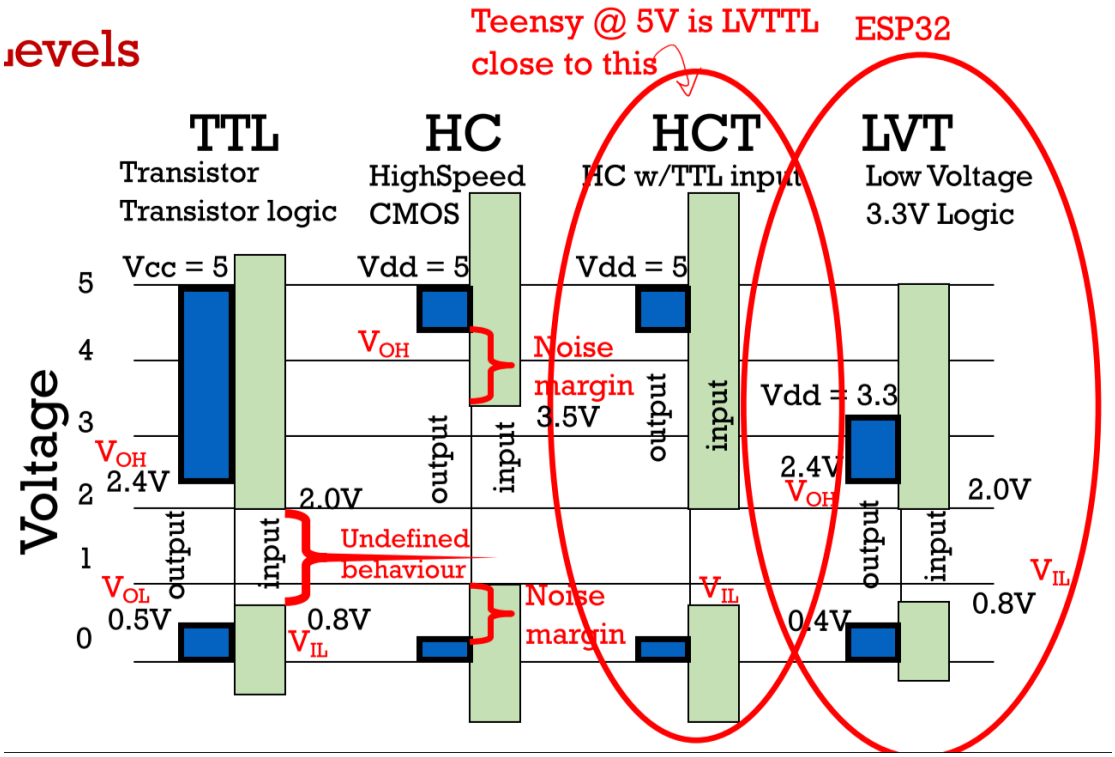

Fig 9 Logical low/high diagram

From table 1 we know that as more light shed on the phototransistors, the output voltage get smaller. And when the resistor connected to the phototransistor is large, the difference between the output voltage with more light and less light become larger as well. The properties of phototransistor TEPT 4400 could be described as below:

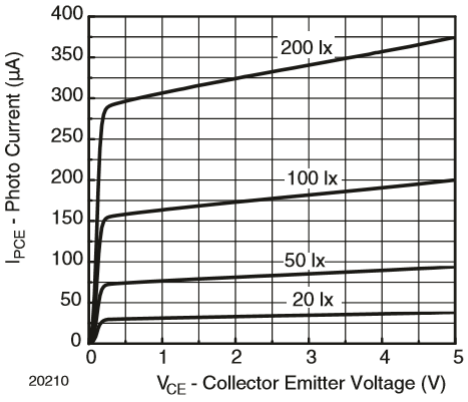

Fig 10 Photo Current vs. Collector Emitter Voltage

From fig 9 we also found out that the when the light is strong, the output voltage is all below 0.3V, which meets the logic low in teensy.

The whole circuit diagram is shown as Fig.7 and the code is in the 2_2_1 folder. The video is in the link:

https://drive.google.com/open?id=1uF-uGAVt0AEB83bWwvjDsZ4L9VW7zA7V

2.2.2 Change the value of the resistors so that the transistor is sensitive enough that waving your hands over the photo transistor (under normal room light) is enough to cause the LED to turn on and off. Submit code and schematic and video of the LED going on and off while you wave your hands at least 20 cm above the phototransistor.

The code is in 2_2_3 folder and the video is in the link. By changing the resistor to 2M， I could wave my hand far more than 1m.

https://drive.google.com/open?id=1DfHnqnoQCYYt2SPgiIkZoQzeVNnCqxcR

The schematic of the circuit is shown as Fig 11.

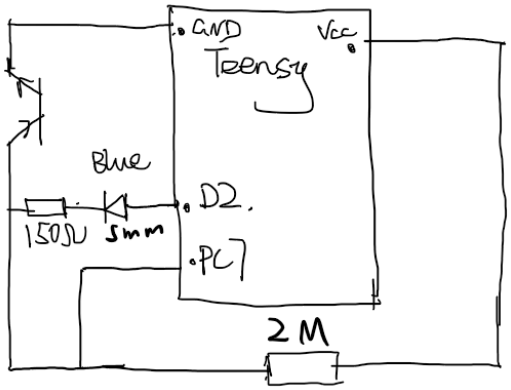

Fig 11 Schematic for Problem 2.2.2