2.1 SWITCHES AND DEBOUNCING AND INPUT CAPTURE

Outcomes: After this first exercise you should be familiar with reading switches on a microcontroller and debouncing them and using input capture subsystems to time events. -

Time expectation: medium -

2.1.1 Get one of the small SPST switch from your kit and 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 pressed. **Submit a schematic of your switch, resistors, Teensy ports and LED. (Code will be submitted later).**

2.1.2 Add to the code from 2.1.1 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 timer3 setting the prescaler to divide by 1024 (use TCCR3B setting CS32:0 to be b101) and print out a time stamp (i.e use `m_usb_tx_uint(TCNT3);`) each time the switch changes state. Notice that when the switch is bouncing with each press (if you depress slowly, there may be more bounces) you may get multiple printouts with short (or same) time stamp values between them.

Add a resistor and capacitor low pass 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 (i.e., 100Hz bounces), but not too large that you distort button presses that occur at roughly 10 times per second (faster than most humans can press). 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 on an oscilloscope or your OscilloSorta. Notice how the signal changes as the frequency goes from below 10Hz to above 100Hz. **Submit OscilloSorta images for each case (ideally two traces one with output of input signal square wave and one with filtered output). Show your calculation for the 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. Video gamers sometimes call this "mashing". Make sure that the switch is debounced, so you only get valid presses. 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.

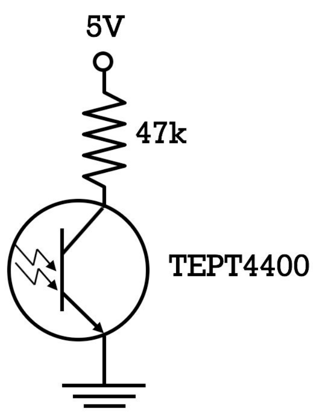
Extra Credit: determine and implement the finest time resolution that you can measure these time presses using input capture 3 assuming the slowest a person will be is 0.5 seconds.

2.2 PHOTOTRANSISTORS

Outcomes: After this section of the lab you should be familiar with using phototransistors to sense light levels and trigger thresholds for digital input high or low. -

Time expectation: medium -

2.2.1



Create the pictured phototransistor circuit and observe the voltage at the point between the transistor and the resistor. This will be the output of the sensing circuit. Observe the voltage at the point between the resistor and the phototransistor on an OscilloSorta as more or less light falls on the transistor (e.g. with the normal room lights on, cover or uncover it with your hand).

Change the resistor value into something 10 x's larger and 10 x's smaller. In your write up answer the following question: **How does the phototransistors behavior change as you change the amount of light that falls on the transistor? Does it reach a valid logic low and valid logic high?**

Connect the output of the sensing circuit to a digital input on the Teensy. Use an external LED attached to a pin configured as an output on the Teensy. Create a program that reads the state of the phototransistor circuit and turns OFF the LED if there is light on the transistor and turns ON the LED if there is no light.

Submit your code and a schematic of your switch, resistors, Teensy ports. Submit a short video showing everything working. See canvas for sample video.

2.2.2: Change the values 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. See canvas for sample video.**

2.3 OPERATIONAL AMPLIFIERS (OPAMPS)

Outcomes: After this section of the lab you should be able to use opamps to amplify weak signals so they are in a useful range .

Time expectation: medium -

2.3.1 Use the function generator from the OscilloSorta (pin 25) to drive the IR LED “LTE-4208” (it is in a clear package) from your kit with a square wave. Find the data sheet for this LED and choose a resistor to put in line with this diode so that the current is close to 30mA but that pin 25 does not have to drive more than 40mA. You may want to verify with a cellphone camera that the IR LED is blinking. **Submit what value resistor you chose for this and the current you expect to be driving.**

2.3.2 Create a photo detection circuit that can show the output of the LED using the opamp in your kit that creates logic level output (for the different states of the LED being on or off) when the IR phototransistor and IR LED are pointed at each other. **Submit a circuit diagram (indicate the output - where the scope is measuring) and snapshots of the OscilloSorta on the output of the circuit. One snapshot of showing that a frequency of about 23Hz (generated from step 2.3.1) is being measured by the trigger line, and one snapshot of 200Hz.**

2.3.3 Connect the detection circuit to a teensy and create a program that can detect frequencies of 23Hz and 200Hz being transmitted and indicate when each frequency is being detected (e.g. turn on a red LED when 23Hz is detected and turn on a green LED when 200Hz is detected). Note that ambient light (room lights) may create 60Hz noise that may confuse your circuit, so, for this part test with the lights turned off or in a dark room.

Submit your code, a circuit diagram. Get checked off by a TA (show during office hours and answer TA questions) that your circuit is working as desired.

2.4 SENSING WITH NOISE

Outcomes: After this section of the lab you should be able to detect a weak signal of known frequency in the presence of 60Hz noise. .

Time expectation: long -

2.4.1 Create a detection circuit that can verify that your overhead lights creates a 60Hz signal. This can be done with a phototransistor, resistor and the OscilloSorta, with or without an opamp. **Get a check off from a TA during evening office hours who will ask you to show the output with and without the lights on. Submit a circuit diagram.**

2.4.2 Create a detection circuit and a filter that can detect the 200Hz signal even when there is 60Hz noise from overhead lights that are on. **Submit a circuit schematic, and a snapshot of the oscilloscope showing the output of the detection circuit. Get a check off from a TA.**

2.4.3 Change the amplification of this circuit to maximize the gain. See how far away you can get from the transmitter and correctly receive both frequencies. Note that the tests should be done while the room lights are ON. **Get a check off from a TA showing that you can exceed 1M. Extra credit will be obtained for exceeding 3 meters.Submit your code, a circuit diagram, and a paragraph description of your approach.**