Spring 2026

ECE 445L Lab 3 Alarm Clock

Alarm clock, LCD, edge-triggered input interrupts, and SysTick periodic interrupts

Table of Contents

Table of Contents 1
Team Size 2
Goals 2
Review 2
Starter Files 2
Required Hardware 2
Lab Overview 2
  Requirements Document 3
Preparation 3
Procedure 4
  Deliverable 1 7
  Deliverable 2 7
  Deliverable 3 7
  Deliverable 4 8
  Deliverable 5 8
  Deliverable 6 8
Lab Checkout 8
Lab Report 8
Hints 8

Team Size

The team size for this lab is **2**.

Goals

- Develop a graphics driver for the LCD that can plot lines and circles
- Design a hardware/software interface for a keyboard or individual switches
- Design a hardware/software driver for generating a simple tone on a speaker
- Measure supply current necessary to run the embedded system
- Implement a digital alarm clock using periodic interrupts

Review

- ebook Section 2.8 or textbook Valvano Section 5.3 on critical sections
- ebook Sections 3.2 3.5 3.6 or textbook Sections 4.5, 4.9, 5.5 on edge-triggered LCD, buzzer
- ebook Section 3.7 or textbook Section 3.4 on developing modular software
- ebook Section 3.8 or textbook Section 3.3 on software style
- ebook Section T.3 or textbook Section 6.2 on periodic timer interrupts

Starter Files

- Starter project:
  - Lab 3 template provided on the Github Classroom repo.

Required Hardware

Parts	Datasheet	Price	Source (**price source)**
EK-TM4C123GXL	EK-TM4C123GXL datasheet	$16.99	**TI**
Sitronix ST7735R Color LCD Adafruit or HiLetGo	ST7735R Driver datasheet	$19.95	**Adafruit** **Amazon**
8Ω or 32Ω speaker	N/A	N/A	EER Checkout Desk
10kΩ resistor	N/A	N/A	EER Checkout Desk
1uF Tantalum Capacitor	N/A	N/A	EER Checkout Desk
Switches	N/A	N/A	EER Checkout Desk
IRLD120 MOSFET or IRLD024 MOSFET	IRLD120 datasheet IRLD024 datasheet	$1.88	EER Checkout Desk Or Mouser, **Digikey**
1N914 Diode	1N914 datasheet	$0.10	EER Checkout Desk, Or **Mouser**

Lab Overview

Labs in ECE445L are extremely open-ended. For Labs 3, 4 and 5 you will be given a requirements document. Your TA is your client or customer. A grade of B can be achieved by satisfying these minimum specifications. To achieve higher grades, you are expected to expand sections 2.1 and 2.5 of the requirements document describing what your system will do. You are free to make any changes to this

document if you achieve the educational goals for the lab. All changes must be approved by your TA. Excellent grades are reserved for systems with extra features and are easy to operate. You will need your LaunchPad and an LCD. From checkout you can borrow speakers, switches, IRLD024 or IRLD120 MOSFET, 1N914, and some resistors for this lab. Clarify from checkout exactly which components checkout expects you to return.

Requirements Document

As always, feel free to adjust the syntax and format of your requirements document as you think appropriate. The goal of the document is to provide a clear and unambiguous description of what the project does. A template of this document is provided in **Lab03Report.docx**.

Preparation

1. Edit the requirements document in the **Lab03Report.docx** to reflect your design. The requirements document is fluid, and we expect it to change as you develop your solution and discover what works and what doesn't. You are allowed to modify the requirements document.
2. Draw a detailed circuit diagram showing all external hardware connections. We expect you to use KiCad (because this is the program with which we will be designing PCBs in subsequent labs). Label all hardware chips, pin numbers, and resistor values. You do have to show connections to the LaunchPad, but not circuits within the LaunchPad itself. You must have in your possession all external hardware parts, but you do not have to construct the circuit. To limit the surge current into the MOSFET, we recommend a 10k resistor between the port pin and the MOSFET gate.
3. For each module you must have a separate header and code file. As stated earlier we expect at least four modules. As part of the preparation, you need to have the software designed, written, and compiled. For the preparation, you do not need to have run or debugged any code. For the modules you have written include a main program that can be used to test it. The SysTick or timer module used to maintain time must be written at a low level, like the book, without calling Tivaware driver code.
4. Write software that implements the digital alarm clock. Figure 3.1 shows the data flow graph.

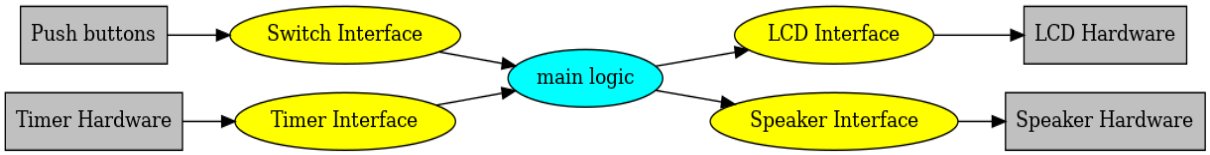

Figure 3.1. Data flows from the timer and the switches to the LCD and speaker.

Figure 3.2 shows a possible call graph of the system. Dividing the system into modules allows for concurrent development, concurrent debugging, and reuse.

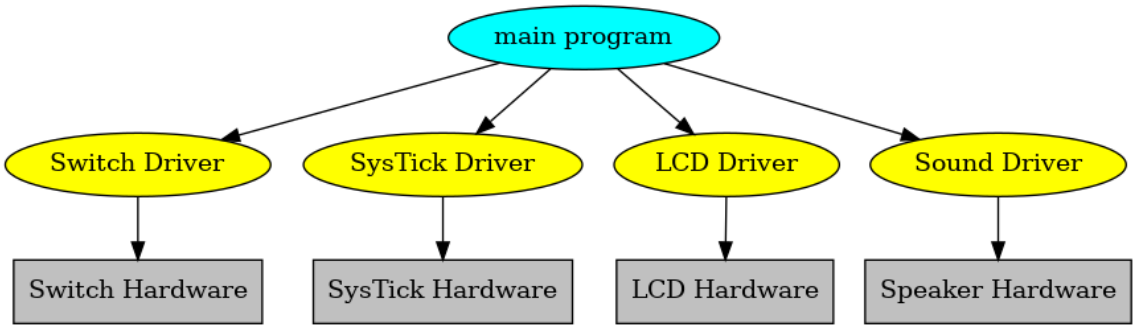

Figure 3.2. A call graph of the four modules used by the alarm clock.

5. Choose a frequency for the alarm sound, f. We recommend choosing a frequency around 1kHz. Let R (e.g., 32Ω) be the resistance of your speaker. Let C (e.g., 1µF) be the capacitance of your ceramic capacitor. Calculate, the time constant of an RC circuit is $\tau = R * C$, in seconds. Calculate the corresponding cutoff frequency is $f_c = 1/(2\pi RC)$. The system should have a cutoff frequency $2 \leq f_c \,/\, f \leq 10$, so it passes the alarm sound, but rejects some of the harmonics of the square wave.

Procedure

1. Build and test any external hardware needed. Debug each module separately. Debug the overall alarm clock. Measure how long it takes to update the graphical time on the LCD. Identify all shared I/O ports and global variables. I.e., document in your software all the permanently allocated variables that have read or write access by more than one thread. Next, consider what would happen if the interrupt occurred between any two instructions of the main program. Remember high priority interrupts can suspend lower priority ISRs. Look for critical sections, and if you find any remove them. Document in your software that each shared object is not critical. During checkout, the TA may ask you to prove that your system has no critical sections.)
2. Record the +3.3V power line rms noise level. The easiest way to measure rms is to use the voltmeter in AC voltage mode. Theoretically, one would expect this rms level to be zero, but the presence of noise will cause the rms value to be somewhere in the range of 0.5 to 5 mV.
3. We expect you to use a real oscilloscope. Use it to measure the speaker voltage when the alarm is sounding with and without the capacitor. I.e., measure the drain pin of the MOSFET (bottom side of the speaker). Figure 3.3a was measured with no resistor between digital output and the gate of the MOSFET, and with no diode or capacitor. Look at the **back EMF**, which is what happens without the 10k resistor, diode, or capacitor. The **dI/dt** when the current is removed is so large the inductance of the speaker produces a 25V spike. Recall the voltage across an inductor is **V=L*dI/dt**. When we interface a device with inductance, we MUST remove the large voltage spikes caused by back EMF. ***Do not try to collect data without the 10k resistor, it will destroy your circuits.***

Figures 3.3a, 3.3b,3.3c, and 3.3d show scope traces at the drain pin of the MOSFET, which is connected to one side of the speaker. The other speaker pin is +3.3V. Current flows when the drain pin is low.

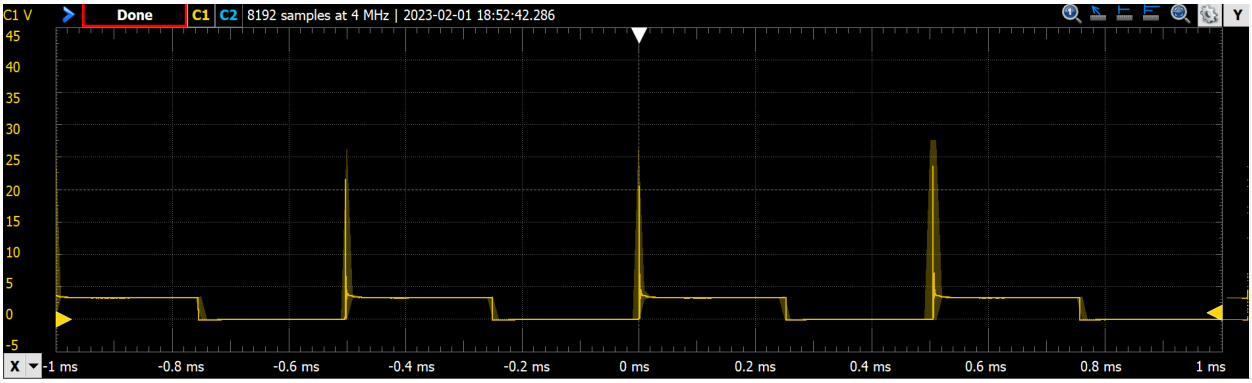

Figure 3.3a. Scope trace of the voltage on the speaker. 32-ohm speaker 0-Ω resistor between pin and gate, and IRLD120 MOSFET.

Figure 3.3b was measured with a 10k resistor between digital output and the gate of the MOSFET, and with no diode or capacitor. The 10k resistor slows the MOSFET down, reducing the dI/dt. Notice the small voltage spikes (4.6V peaks) that occur when the current is turned off. Use the scope to verify the sound frequency is 2 kHz.

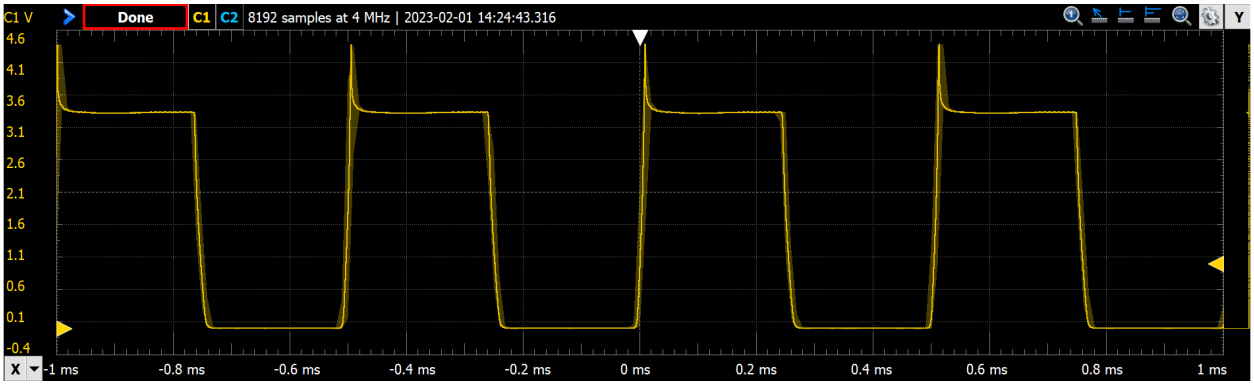

Figure 3.3b. Scope trace of the voltage on the speaker with 10k at gate.

Decide whether you wish to use either a diode or a capacitor to remove the back-EMF spikes. Figure 3.3c was measured with a 10k gate resistor and a 1uF capacitor. Notice the small voltage spikes are gone and the edges are round.

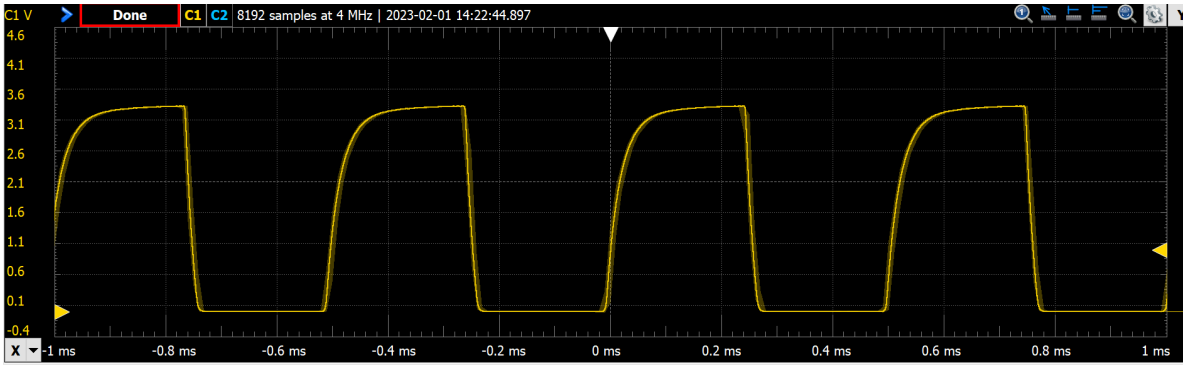

Figure 3.3c. Scope trace of the voltage on the speaker with 10k at gate and 1uF a capacitor across the speaker.

Figure 3.3d was measured with a 1N914 snubber diode. Notice the voltage spikes are smaller but not totally gone.

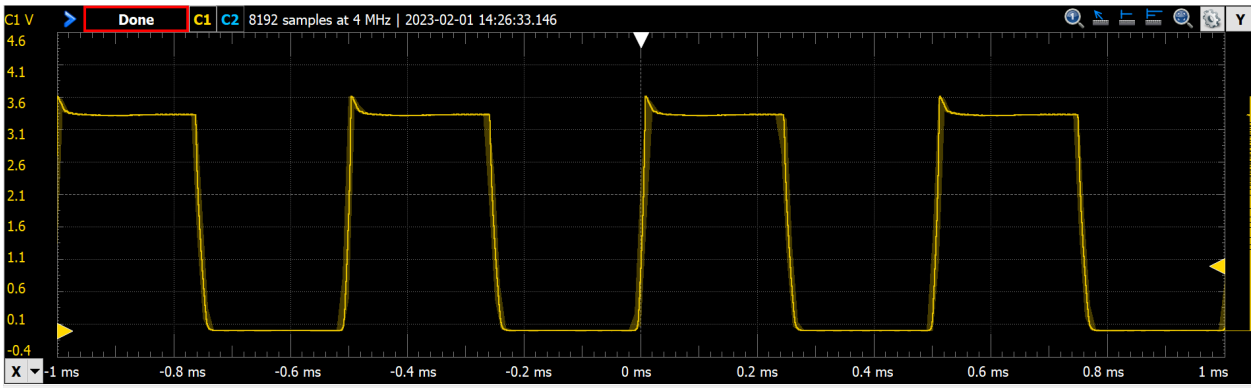

Figure 3.3d. Scope trace of the voltage on the speaker with 10k at gate and a 1N914 snubber diode across the speaker.

4. Measure the 3.3V supply current. Remove the jumper on the LaunchPad (see Figure 3.4) and connect a DC current meter across the pins. Double check the connections before turning it on. If you are at all unsure about this measurement, ask your TA for help. Measure the required current to run the alarm clock. Take a measurement with and without the alarm sounding. *Remember these numbers; they will be valuable when you design your project in Labs 6 and 7.*

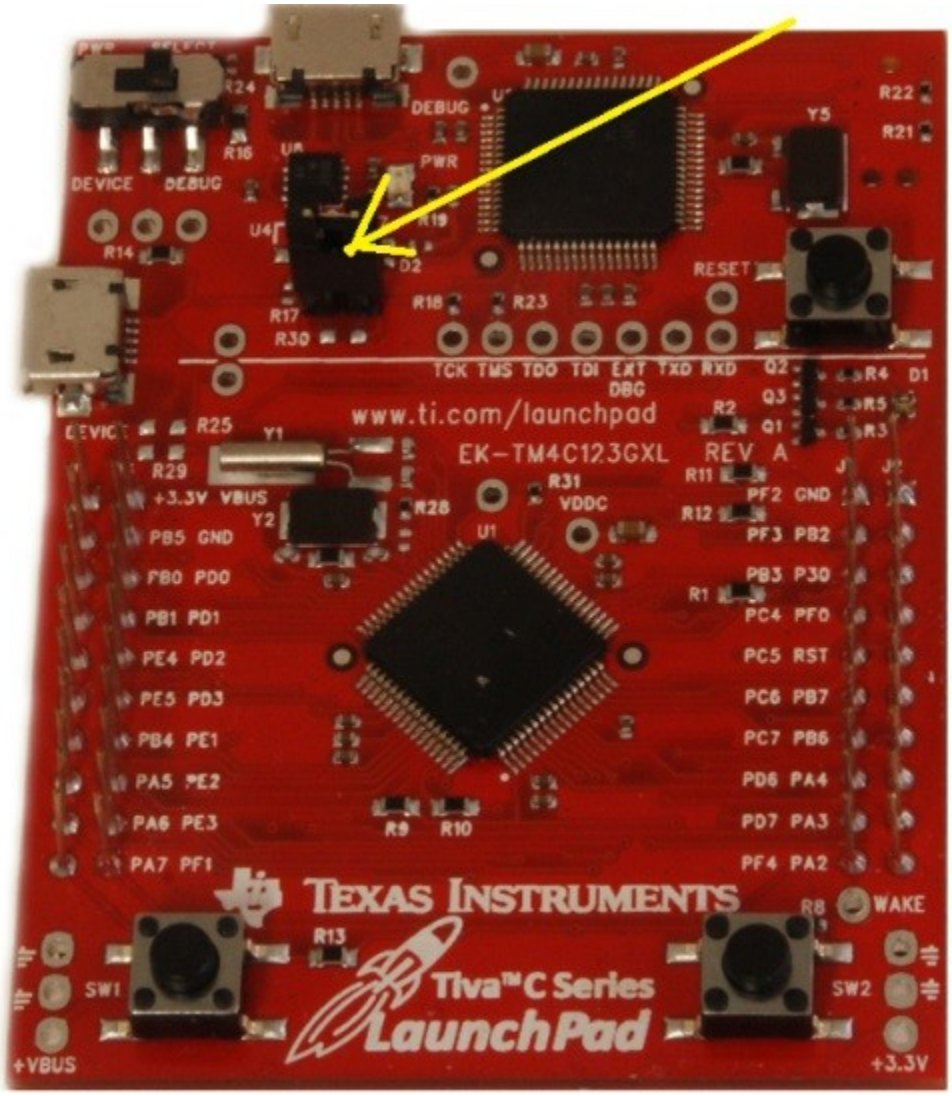

Figure 3.4. Use a current meter to measure current required to run the clock.

Deliverable 1

Draw the electrical circuit you used to create the alarm clock. Put either a diode or a capacitor in parallel with the speaker. If you use a polarized tantalum capacitor, orient the + and - pins with the direction of the current flow. The diode on the other hand must be oriented with the + and - pins opposite with the direction of the current flow.

Deliverable 2

Measure how long it takes the LCD graphics to update.

Deliverable 3

Show the RMS noise level on the 3.3V with and without the alarm sounding. Using a voltmeter in AC mode, you should get a value of between 0.5-5mV.

Deliverable 4

Measure the voltage versus time of the drain pin of the MOSFET, without capacitor, and use it to determine the current through the speaker. Measure the frequency of the sound. Place a picture of the scope trace like **Figure 3.3b** into your lab report, either a photo or digital downloaded image

Deliverable 5

Measure the voltage versus time of the drain pin of the MOSFET, with the capacitor or the diode. Place a picture of the scope trace like **Figure 3.3c** or **Figure 3.3d** into your lab report, either a photo or digital downloaded image

Deliverable 6

Show the system current with and without the alarm sounding. This is measuring the current through the entire system. You can do this by connecting the lab DC power supply (set to 5V and 500mA) to the TM4C's VBUS and GND.

Lab Checkout

The lab checkout is performed during the M/T lab session.

Demonstrate all the functions of your alarm clock including the required ones as well as any extra features. Additionally, show that your digital alarm clock is stand-alone by turning the power off, then on. The digital alarm clock should run (the time will naturally have to be reprogrammed) without downloading the software each time.

Lab Report

The lab report shall be submitted by the Friday after the second lab section.

You should complete the Lab03Report.docx file with your data and answers then submit the completed file to Canvas.

Hints

1. The requirements document should change a couple of times during the lab as you determine features.
2. You can interface an 8Ω or 32Ω - speaker to an output port using an NPN MOSFET like the IRLD120. A 10kΩ resistor between digital output pin and gate reduces current surges but does not affect loudness. Loudness is determined by the voltage drop across the speaker. Connect the source to ground, and the drain to one side of the speaker. Connect the other side to +3.3V. The maximum I_{DS} of the transistor must be larger than 3.3V/8Ω or (3.3V/32Ω). The speaker has inductance, but the MOSFET includes an internal diode to remove back EMF when the transistor switches off. If you toggle the output pin in the background ISR, then sound will be generated. Loudness is determined by the voltage drop across the speaker. From Figure 3.3a, we see the MOSFET drain voltage is about 0.5V when active. So, the voltage drop will be 3.3V-0.5V = 2.7V
3. You must be careful not to let the LCD show an intermediate time of 1:00 as the time rolls over from 1:59 to 2:00. You must also be careful not to disable interrupts too long (more than one interrupt period), because a time error will result if any interrupts are skipped.

4. You may use 32-bit or 64-bit timer modes on the TM4C microcontrollers. However, it is good practice to refer to the errata for the microcontroller you are using. The errata describe bugs and flaws not listed in the data sheet.
5. If you use the on-board switches, then you must activate the internal pull-up resistors. You will set the corresponding bits in the GPIO_PORTF_PUR_R register. These on-board switches are simply SPST switches to ground. When the switch is pressed, the signal goes to 0V (ground). When the switch is not pressed, the internal pull-up makes the signal go high (3.3V.) Furthermore, coming up out of a reset PF0 is locked, and thus if you use PF0 you will need to unlock it.
6. Learn to use KiCad. You will need to be proficient with this application during Labs 6 and 7. Using it now for simple circuits will be an efficient use of your time.
7. If you use edge-triggered interrupts, build an analog filter to debounce each switch. Set R1=100, and R2=100k to create the negative logic switch. Choose R2 and C1 so the time constant (τ=R2*C1) is around 10 ms. Test the circuit with a scope before connecting to the microcontroller.

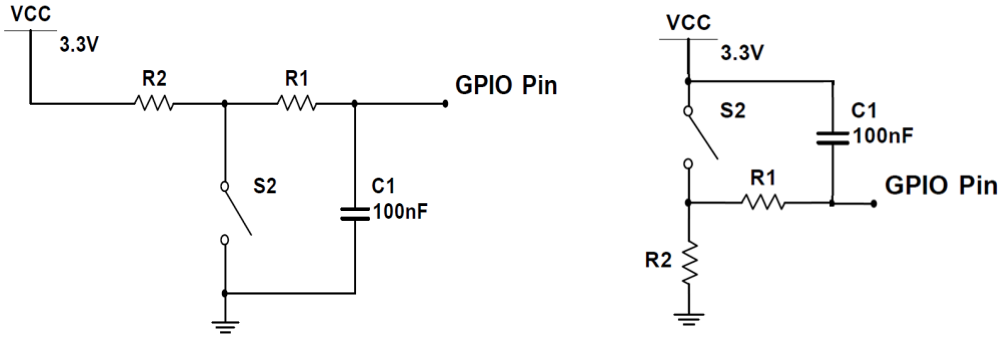

Figure 3.5. Switch debouncing with a capacitor.

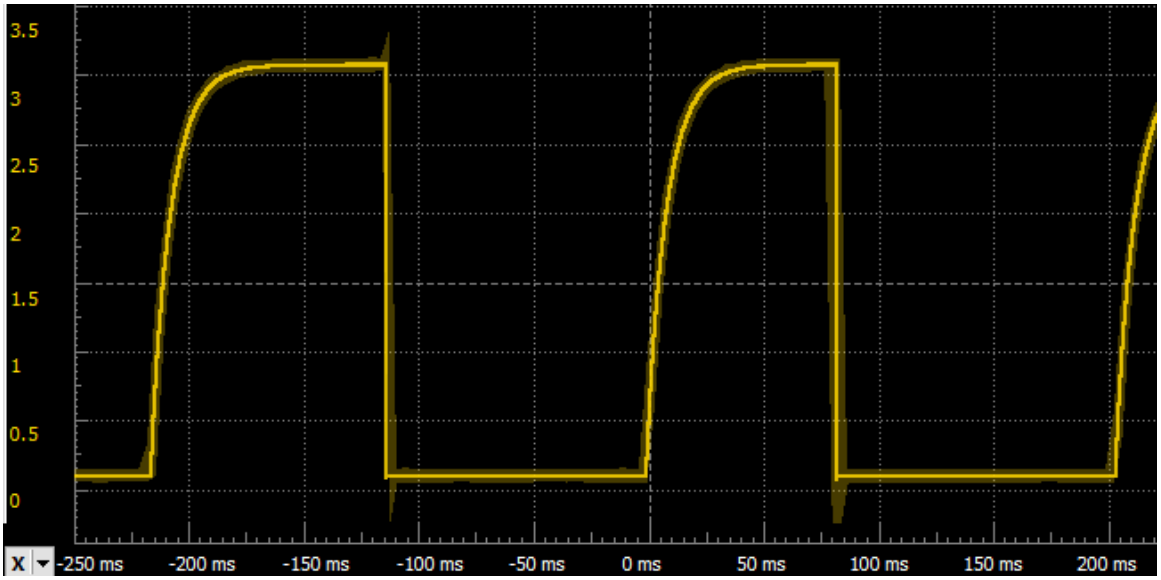

Figure 3.6. Waveform for negative logic switch interface.

8. A better method to debounce edge-triggered interrupts is to use a second timer (see **EdgeInterruptDebounce_4C123**)