Lab 6: Interfacing Switches and LEDs

Preparation:

- Before you start the lab, ensure that you have completed Lab 2 by successfully interfacing QTR-8RC line sensor to MSP 432 LaunchPad and tested it. You will require LEDs, resistors, push button switches and breadboard along with MSP 432 LaunchPad to complete this lab.

Reference Materials:

- MSP 432 Launchpad User's Guide
- MSP432P401R Datasheet
- B3F-1052.pdf Switch Datasheet
- HLMP-4700.pdf LED Datasheet

Purpose:

The purpose of this lab is to interface a switch and an LED to MSP 432 LaunchPad. You will build the circuit on breadboard and hence verify whether its operational by running the driver program. Finally, you will design your own 'Window Intruder Alarm System'.

Procedure:

In this lab, you will design a 'Window Intruder Alarm System' as shown in figure 1.

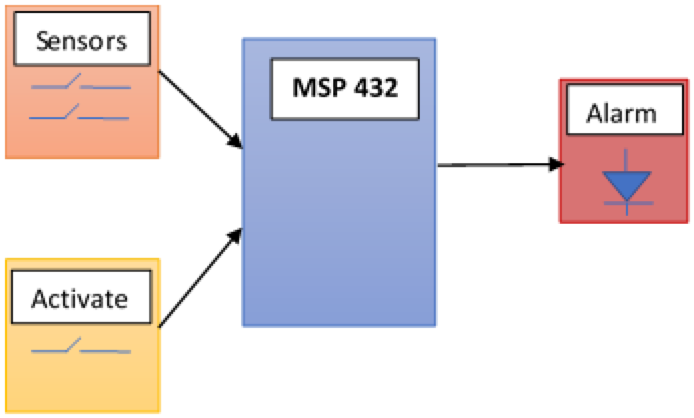


Figure 1

The system consists of three switches out of which two acts as 'sensors' and last one acts as 'Activate' switch interfaced to MSP 432 LaunchPad. The alarm is signified though an LED connected to the LaunchPad. When 'Activate' switch is pushed, the system gets activated and detects for sensor switches. The sensor switches are said to be 'secured' when are closed and unsecured when open. On the other hand, if the system is not activated (by pushing activate switch) it does not monitor the sensor switches and hence the system is said to be 'disarmed'. The alarm is indicated by flashing a red LED at 5Hz (100ms ON and 100ms OFF).

Connect the switches and LED to breadboard and interface it to MSP 432 LaunchPad. The system should work as shown in the truth table 1 shown below.

Activate Switch	Window 1 Sensor	Window 2 Sensor	Alarm (LED)
OFF	X	X	OFF
ON	Open	Open	Flash
ON	Open	Closed	Flash
ON	Closed	Open	Flash
ON	Closed	Closed	OFF

Truth Table 1 Switch

Interface:

Before you connect all 3 switches, start by interfacing one switch to MSP 432 LaunchPad. Figure 2 shows a possible way to connect the switch to the microcontroller. Further, connect an internal/external pull-down resistor to pull the pin P5.0 to logic 0 when the switch is open.

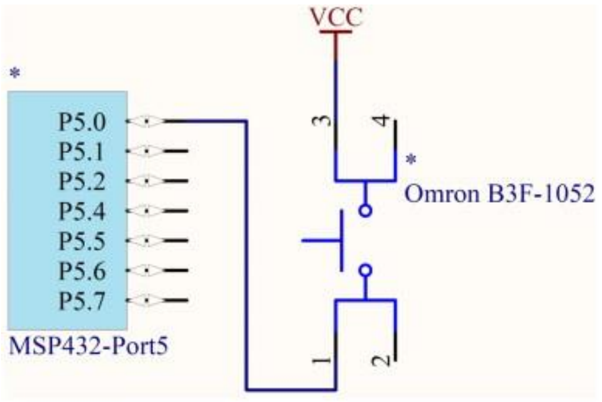


Figure 2

Now, write a software driver to interface one switch to pin P5.0 based on steps below.

- Set bus clock at 48MHz.
- Configure pin P5.0 as GPIO by resetting bit 1 registers SEL0 and SEL1.
- Now, make P5.0 as input.
- Next, enable pull resistor on P5.0
- Finally, read the switch status continuously in a while loop.

Next, using a volt meter, measure the voltage across the pin P5.0 when the switch is pressed (3.3V) and when its not pressed (0V). You should eventually modify the hardware and software to accept 3 inputs from all three switches.

LED Interface:

The first step here is to find the operating point of the LED by determining (V_f, I_f) from the LED data sheet and calculate the appropriate resistance value required to be interfaced to the LED using the formula

$$R = \frac{3.3 - Vdd}{Id}$$

Where V_{dd} and I_d is the forward voltage and current through the resistor R.

Decide an appropriate pin on Port 5 to connect the LED and build the circuit as shown in figure 3. Here we are using the same pin P5.0.

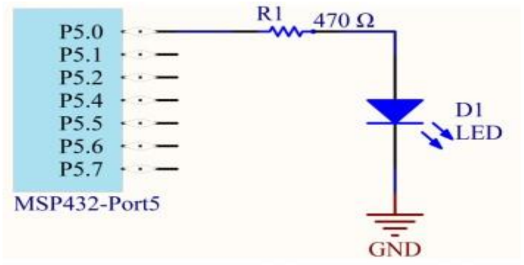


Figure 3

Perform LED measurements with different resistance values Ex. 330Ω, 470Ω, when the software outputs high (logic 1) and calculate the current through and voltage across the LED for the circuit built. Next, write a software driver to flash the LED based on the steps below.

- Set bus clock at 48MHz.
- Configure P5.0 as GPIO by resetting bit 1 of registers SEL0 and SEL1.
- Next, make P5.0 as output.
- Turn on the LED by setting bit 0 of OUT register.
- Turn OFF the LED by resetting bit 0 of OUT register.

Use function abstraction by separating what LED does (Init/on/off/toggle) from how it works (Pin P5.0). Run the software by single stepping unit the LED is on. Measure the voltage across the resistor and that across LED. Repeat the steps and perform measurement when the LED turns off. Calculate the current through the resistor which should be equal to that through LED. Now, compare the actual (V_f, I_f) operating point of the LED with the expected values calculated at the time of design.

LED Toggling:

Here, the objective is to flash the LED 5 times a second. You can do this by writing a delay function Wait1ms() by using nested for loops. Though there are not very accurate, you'll end up flashing the LED somewhere between 4 to 6 times a second. Use a logic analyzer to verify that LED flashes at the required rate. To activate logic analyzer to visualize Port 5, execute

```
TExaS_Init(LOGICANALYZER_P5);
```

Window Detector Alarm System:

We'll use 3 pins P5.0, P5.1 and P5.2 to interface switches to the LaunchPad while pin P5.4 to interface to the LED. Design the circuit as shown in figure 4 below.

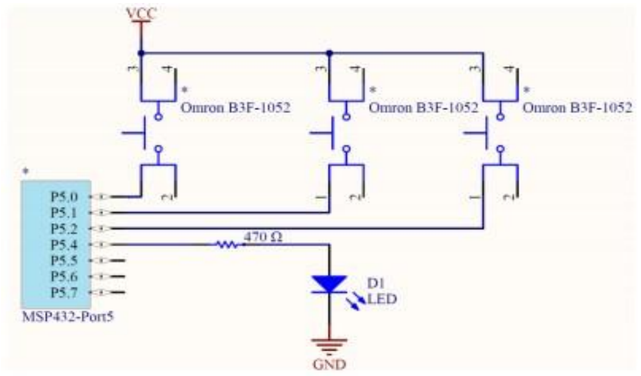
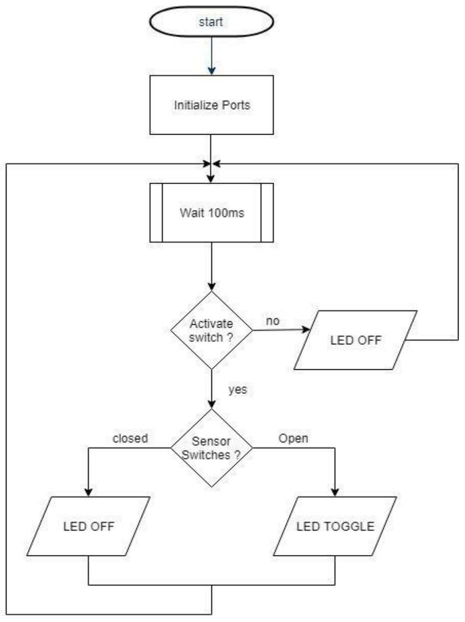


Figure 4

Develop software driver based on the steps below to configure the alarm system.

- Configure pins P5.0, P5.1, P5.2 and P5.4 as GPIO by resetting bit 1 of registers SEL0 and SEL1.
- Make the pins P5.0, P5.1 and P5.2 as input and P5.4 as output.
- Start the system with LED OFF.
- Wait for about 100ms.
- Monitor the 'sensor' switches. If 'activate' switch is pressed and one/both sensor switches are open, the alarm must be activated by flashing the LED at 5Hz else the LED must turned OFF.
- Repeat last 3 steps indefinitely.

The flow chart is shown below.



The structure of this lab is to build and test individual sub-system and then integrate switches and LED interfaces to implement the Window Detector Alarm System. Test the overall system by single stepping through the software and verify its operation for all cases mentioned in table 1.

Questions:

1. How would you make the LED brighter?
2. Modify the existing system to include a second green LED to indicate if the intruder system is activated or not. Design the circuit and write the software driver for the same.