**Arduino Digital input with de-bounce**

Introduction

We'll use various methods to query a button for input to a microcontroller. A mechanical switch is biased to create a HIGH or LOW input to a programmable device but the mechanical aspect of the switch means that there could be "chatter" as it transitions between HIGH->LOW or the other way.

* Create a simple polling routine that reports the change status of an attached switch via serial monitor output.
* Observe raw switch bounce via O-scope and Serial monitor messages (if access to O-scope)
* Mitigate switch bounce with analog components and observe the effect via 0-scope and serial monitor messages
* Mitigate switch bounce with software and observe the effect with serial monitor messages
* Create an interrupt driven switch service routine and observe processing.

MATERIALS

* Laptop or EE101 Lab computer
* Your Lab notebook
* Arduino Programming book (Blum)
* Arduino IDE installed and working on your laptop or EE101 Lab computer
* MakerSpace Kit

Exercises

Startup video. <https://streaming.uvm.edu/private/videos/ymCg1tP/>

1. **Digital input displayed on serial port: Level Sensitive.**

Description: Create polling loop that tests in the output of a mechanical switch and sends a simple message to the serial monitor when the switch state changes ( low to high or high to low). Keep the messages very short. (i.e. 0 for low and 1 for high)

Procedure:

* Its OK to use examples in the Arduino IDE to get started and to solve problems.
* However… adhere to the requirements. No extra stuff in your program other than what is called for in the requirements. Ex. The “StateChangeDetection” example has input state detection and serial message output. (OK) It also has button press counting, LED control and a delay for debouncing. (not OK). As we move through the exercises, you can incorporate more features from examples.
* Attach one of your pushbuttons along with a Pull-up resistor ( 10K) to RoboRed I/O pin 5.
* Write or modify a polling algorithm to look at pin 5 and report a state change via serial monitor. Also add the onboard LED attached to pin 13. The logic should follow the flow chart shown in Figure 2.

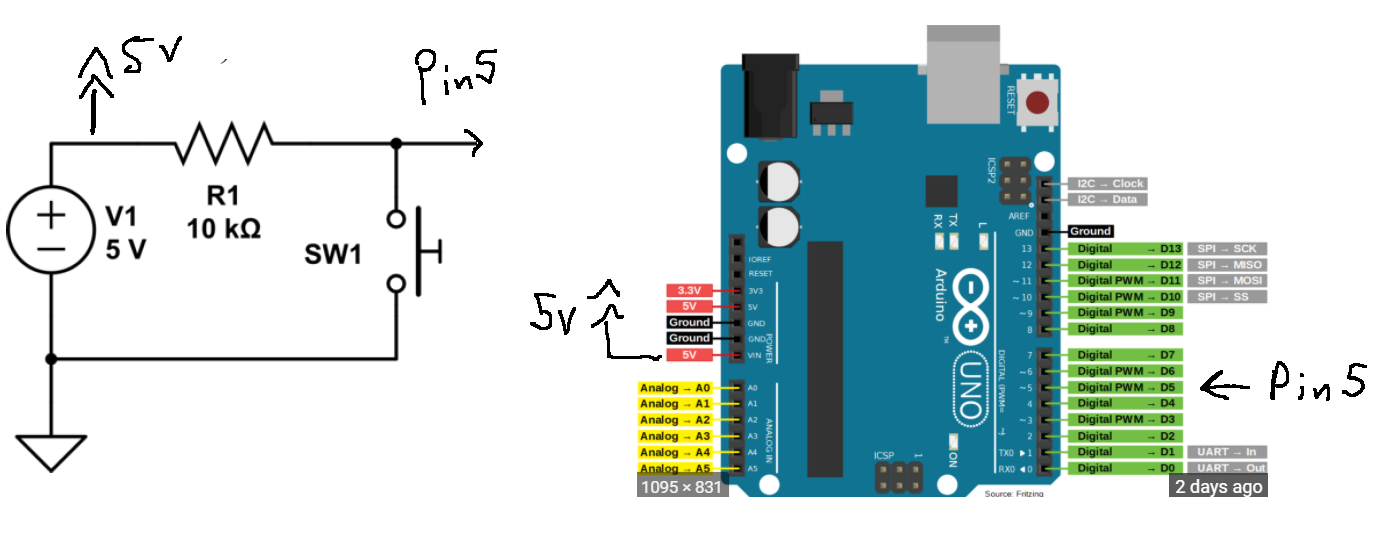


Figure 1. Diagram of Arduino UNO with active low diode output attached to pin5.

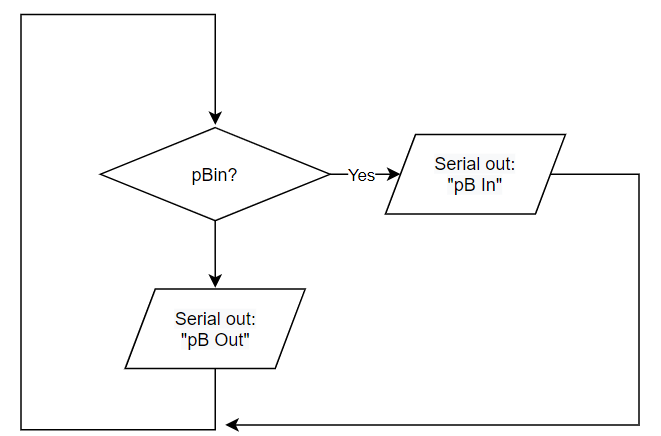


Figure 2. Flow chart for a level sensitive action on an LED or serial output message. The message is ONLY sensitive to the state of the button.

1. **Debounce a button using software delays. Edge sensitive.**

Description: The edge sensitive operation of an LED output or serial message changes state ONLY when an edge is encountered. That is, the LED changes state when the button is pushed in (or released). It is staid to be edge sensitive. In this mode, the LED is subject to button chatter because many edges can be encountered as the button makes ore breaks contact. Reducing this chatter is what we’ll do with delays.

Reduce the switch bounce observed in Exercise1 by software means so each press and release of the switch results in two serial monitor messages (one high to low transition and one low to high transition).

Procedure:

* Modify your program from part 1 to be edge sensitive as shown in Figure 3. Note that it uses a while loop to keep the output in the state as long as the button is held in.

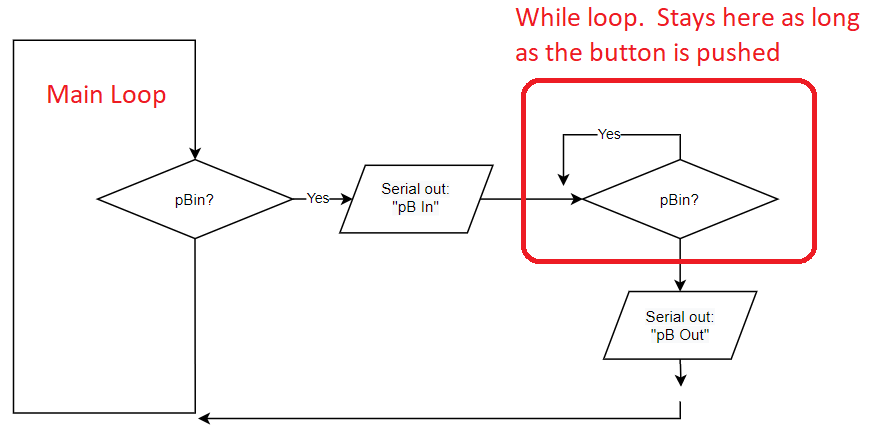


Figure 3. Edge sensitive flow with NO DEBOUNCE added.

* Modify your program to debounce the switch input via software. Note serial monitor output.
* Note that figure 4 shows a flow chart for how this could be done.
  + Demo this exercise to your TA. Make sure you satisfy all requirements above. Make sure you don’t see any debounce.

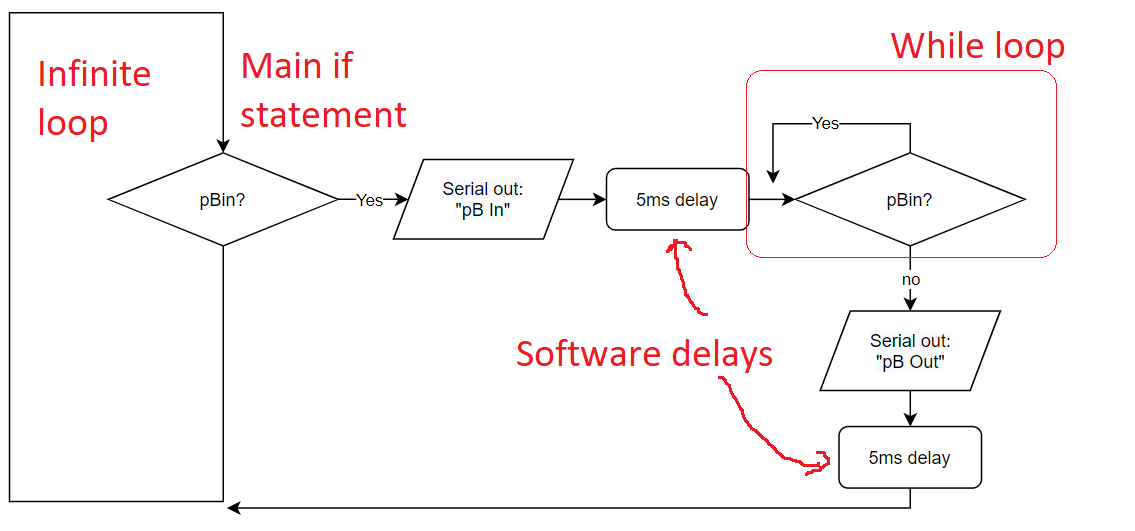


Figure 4. flow chart of a debounced button using software delays

1. **Using an interrupt**

Description: Implement switch processing via interrupt. Make an LED blink on and off, on time for each switch press/release cycle. On time is approximately 200ms. Research the interrupt on the web to find the code needed to be included and the functions available.

Procedure

* Modify the last program to add an Interrupt Service Routine (ISR) that is triggered by a high to low state change of your switch attached to the INT0 input (Digital input 2).
* Move your Digital input from D5 to D2. You will have to move the 5ms delay into your ISR. (move the pin from D5 to D2 on breadboard)
* In the main loop, blink the LED attached to Digital IO 13 (on your board) if the complete on-off switch cycle is satisfied.
* Figure out a delay mechanism to illuminate the LED for 200ms +- 5%