UNIVERSITY OF NORTH CAROLINA AT CHARLOTTE
Department of Electrical and Computer Engineering

EXPERIMENT 2 – SWITCH DEBOUNCING

OBJECTIVES

This experiment will highlight one of the problems encountered in digital systems design, that of switch bouncing. In this laboratory an example is provided to illustrate the full effect of this problem and its solution, a switch-debouncing circuit.

MATERIALS/EQUIPMENT NEEDED

DC Voltage Source (capable of 10 Vdc)
Oscilloscope
(2) SPST switches
(1) SPDT switch
(3) Resistors: 10kΩ
(4) Resistors: 330Ω
(1) 74161 chip: Up counter
(1) 7400 chip: Quad 2 input NAND

INTRODUCTION

An important (but often overlooked) concern in digital design is switch bounce. The basic mechanisms involved are related to the mechanical design of the switch and the large electric fields that develops when the contacts are very close but not touching. The result of this situation is arcing between the contacts until they finally settle down and make permanent contact. In many design situations, this arcing (or bouncing as it is more often referred to) is of no concern to the system operation. In other situations, however, this switch bouncing can cause seriously undesirable results.

As an example where we see the full effect of the switch bounce problem, consider the circuit illustrated in Figure 2-1. In this circuit, we use a simple push button to provide the clock input signal to a binary counter and another push button switch to provide the clear input signal. The counter is a 74161, 4-bit, binary "up" counter, which is leading edge-triggered with an active low asynchronous clear and an active low synchronous parallel load. To disable the parallel load this input is connected to V_{cc}, +5 V for this chip. Observe that a 10kΩ resistor is used to "pull-up" the clock, and clear inputs to the counter.

With these switches in the open position a logical 1 is applied and in the closed position a logical 0 is applied. It is good design practice to always pull-up (to a logic 1) or pull-down (to a logic 0) critical unused or unconnected inputs. The unwise alternative would be to let such inputs "float" and expect that they will always assume and remain at the desired level. When the Clear button is pushed, the clear input which is active low is pulled to ground (logical zero), and the counter resets to the all zeros state. Since the clear input to the counter is asynchronous, clearing happens automatically without a clock pulse. This is the appropriate behavior by the system and we see no effects of switch bouncing (repeated low going pulses to this input would do nothing more than clear the counter again and again).

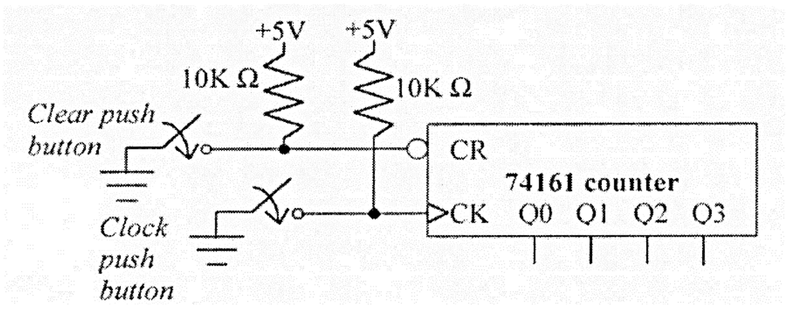


Figure 2-1 Example circuit

The desired behavior of the counter with respect to the Clock pulse is that no change in the output (Q_0 through Q_3) is observed on the trailing edge when the button is pressed in, but each time the button is released, causing a leading edge, the counter will increment by 1. As illustrated in Figure 2-2, if switch bounce or arcing is present, the desired operation is prevented and multiple leading edges will cause the counter to increment by an unknown amount as the clock input bounces back and forth between low and high. When final contact is made the clock remains at a logical zero until the switch is opened; but then, arcing begins and continues until the contacts are far enough apart to prevent arcing. Thus, we see multiple rising edges at both the closing and opening of Clock button, and as a result, an unknown accumulation of counts by the counter.

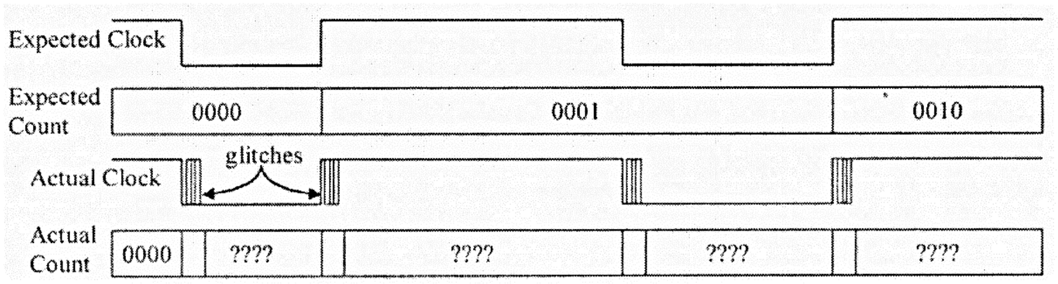
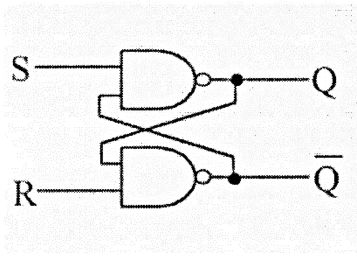


Figure 2-2 Clock signal and output response

The solution to this problem is a switch de-bouncing circuit, which makes use of an SR latch constructed from cross-coupled NAND gates as shown in Figure 2-3, along with its associated state table. By inserting the SR latch into the Clock push button circuit, as shown in Figure 2-4, we can overcome the bouncing problem by making use of the Set, Reset, and Storage states of the SR latch. Notice that we have also changed the type of switch from a single-pole, single-throw switch to a double-pole, single throw switch, and thus, have needed to add an additional pull-up resistor. The double-pole switch ensure that the S and R inputs to the SR latch will both be at a logical 1 (putting the SR latch in the storage state) when no contact is made with either pole. This is the case while we are in the process of pushing or releasing the switch.



S R	Q $\overline{Q}$	state
1 1	Q $\overline{Q}$	storage state
1 0	0 1	reset state
0 1	1 0	set state
0 0	1 1	indeterminant

Figure 2-3 SR latch schematic and state table

Comparing the action of the switch to the state table for the SR latch we can see how the switch debouncing circuit works to prevent glitches in our clock signal. Assume that the switch is normally connected to the S (set) input of the SR latch as illustrated in Figure 2-4. As a result, the S input is pulled to a logical 0 and the Q output to a logical 1, which is the Set state of the SR latch. When the clock button is pushed, the break in contact takes the SR latch inputs from S=0 and R=l to S=1 and R=l, corresponding to the storage state where the Q output continues to be a logical 1. Glitches at the S input of the SR latch will take the latch between the set state and the storage state, keeping the Q output at a logical 1. As the switch arm moves toward the contact connected to the R input, the latch remains in the storage state with Q = 1 until it encounters the first arc between the switch arm and the R contact. With this first arc, the inputs to the latch change from S=1 and R=1 to S=1 and R=0, corresponding to the reset state which produces a logical 0 at the Q output. Subsequent glitches take the RS latch between the storage state and the reset state keeping Q unchanged. As a result, we see a clean falling edge of the clock input to the counter as illustrated by the Expected Clock in Figure 2-2.

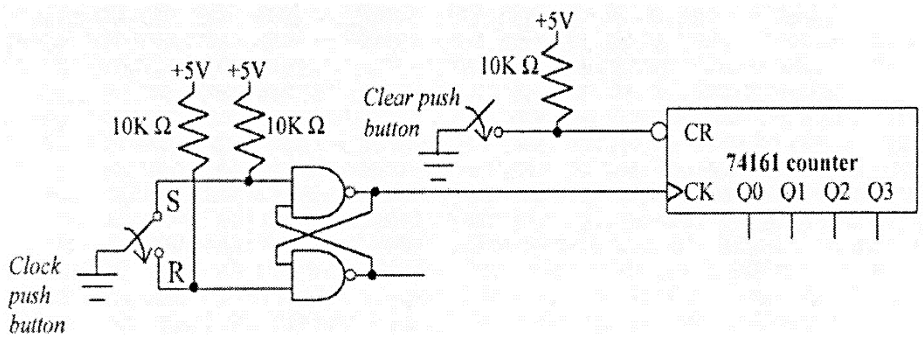
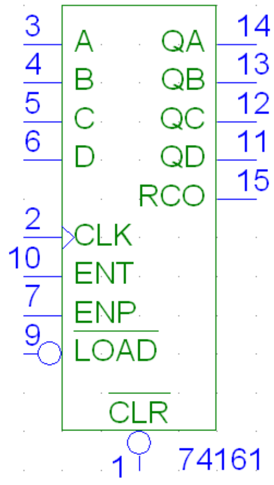


Figure 2-4 Switch debouncing circuit

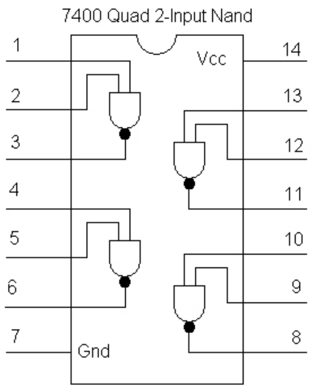
When the button is released, the SR latch moves between the reset and storage states as arcing glitches continue until sufficient distance is achieved and the arcing stops. As the switch arm nears the contact connected to the S input, arcing begins, and the latch moves between the set and storage states with the first glitch producing a logical 1 at the Q output. This transition in Q produces a rising edge of the clock signal and the counter is incremented once and only once until the next press and release of the clock push button. Note that the double-pole switch at the inputs of the SR latch prevents the latch from entering the indeterminate state where both S and R are 0 and both the Q and Q bar outputs are forced to a logical 1. In this situation the final state of the latch is indeterminate and depends on what is referred to as a race condition. Furthermore, a conflict of this type could, in some cases, result in circuit damage.

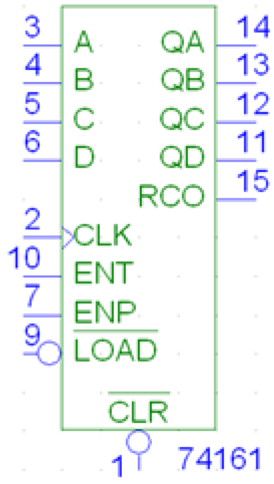
PRELAB

1. Given the logic diagram shown in Figure 2-1, along with the appropriate data sheets for the 74161 (binary counter) integrated circuit, **draw a schematic for the circuit**. Most importantly, be sure to label the IC pin numbers since you will be using these schematics during the lab session.



2. Given the logic diagram shown in Figure 2-4, along with the appropriate data sheets for the 74161 (binary counter) and 7400(quad 2-input NAND gates) integrated circuits, **draw a schematic for the circuit**. Most importantly, be sure to label the IC pin numbers since you will be using these schematics during the lab session.





Show your pre-lab schematics to the lab TA at the beginning of the lab session. Note that the control inputs LOAD and CLR are active low. Thus, they must be connected at a logical high (5V) to be inactive.

PROCEDURE

1. Show your pre-lab schematic to the lab TA at the beginning of the lab session.
2. Construct the circuit without debounced switch (Figure 2-1) on the prototype board. Connect the counter outputs to LEDs on the prototype board in order to monitor state of the counter during the exercise.
3. Clear the counter by pressing the Clear push button and verify that the counter goes to the all 0s state.
4. Press the Clock push button but do not release the push button and record the count value with the push button depressed along with the value you would expect for the counter if there were no switch bouncing problems.
5. Release the Clock push button and record the count value after the push button is released along with the value you would expect for the counter if there were no switch bouncing problems.
6. Clear the counter and verify that it has reset to the all 0's state.
7. Then press and release the Clock push button and record the count value. Repeat this process of clearing the counter, then pressing and releasing the Clock push button a number of times (at least 5 times) while recording the count value each time. Try pressing and releasing the button slowly as well as fast to see if there is any difference in the count values. Take the average of the count value obtained for the press/releases of the Clock push button.
8. Construct the switch debouncing circuit and integrate it with your counter to obtain the circuit in Figure 2-4. Repeat Steps 3 through 7 for the debounced clock circuit recording values as indicated in each step.
9. Before you disassemble your circuit and leave the lab, show your lab TA that your debouncing circuit works properly.

DATA/OBSERVATIONS

Table 2-1: Results without debouncing switch

Trial	**Expected Count Value**				**Measured Count Value**			
	LED 1	**LED 2**	**LED 3**	**LED 4**	**LED 1**	**LED 2**	**LED 3**	**LED 4**
1								
2								
3								
4								
5								
6								

Table 2-2: Results with debouncing switch

Trial	**Expected Count Value**				**Measured Count Value**			
	LED 1	**LED 2**	**LED 3**	**LED 4**	**LED 1**	**LED 2**	**LED 3**	**LED 4**
1								
2								
3								
4								
5								
6								

INSTRUCTOR'S INITIALS:		DATE:	

POST-LAB

1. Correct any mistakes in your two schematics that you encountered during the construction and operation of your two circuits (with and without the switch debouncing circuit).
2. Use the data you took during the lab session (in tabular form) to write a brief discussion of the effectiveness of the switch debouncing circuit based on the data.
3. Given that the debouncing circuit is somewhat expensive in terms of hardware (2 NAND gates, 2 resistors, and a double-pole, single throw switch), describe applications where you would require switch-debouncing circuits as well as applications where you would not need to include the additional hardware for switch debouncing (in other words, applications where you can tolerate switch bouncing). Note, you cannot use the clock and clear inputs of our lab as example applications; instead you need to think of other examples.
4. Finally, evaluate this lab exercise in terms of its value added to your understanding of digital logic circuits. Include any suggestions you have to improve this lab for subsequent semesters.

Be sure to include all items from the post-lab exercise above in your written lab report.