6

Gates and Flip-Flops

This lab continues our exploration of transistor switching circuits to include CMOS logic gates, latches, and flip-flops. We will explore simple shift-register and counter circuits using small arrays of flip-flops. We will also demonstrate a simple scheme for digital-to-analog conversion.

Pre-lab Preparation

Before Coming to the Lab

Read through the lab experiment to familiarize yourself with the components and assembly sequence. Before coming to the lab, each group should obtain a parts kit from the ECE Shop.

Parts List

The ECE2 lab is stocked with resistors so do not be alarmed if you kits does not include the resistors listed below. Some of these parts may also have been provided in an earlier kit.

Laboratory #6	
Logic Gates and Flip-Flops	
Qty	**Description**
1	CD4007 CMOS pair/inverter
2	CD4011 Quad 2-input NAND gate
2	CD4013 Dual D flip-flop
2	CD4093 Quad 2-input NAND Schmitt
4	Tactile pushbutton switch, SPST PCBPB1
4	Red LED 20mA
4	Ceramic capacitor 0.01uF (10nF)
4	470-Ohm 1/4 Watt resistor
4	1-KOhm 1/4 Watt resistor
3	10-KOhm 1/4 Watt resistor
5	20-KOhm 1/4 Watt resistor

Background information

Suggested background reading:

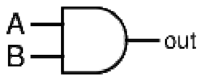
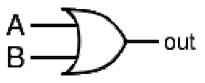
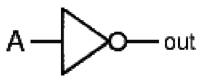
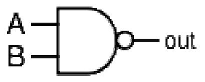
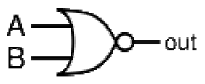
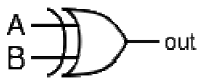
- Hands-on Electronics, Chapters 10-13

Digital Logic Functions

Digital electronics is largely based on two-state or binary logic. The two logic states are designated “0” (low) and “1” (high) and defined electronically by fixed voltage levels. The low-state is usually the reference or ground potential in the circuit, and the high-state is some positive voltage such as +5 V.

A logic “gate” takes one or more logic-level inputs and produces a single logic-level output. The output is also a logic level, so the output of one logic gate can connect to the input of one or more other logic gates. Table 6-1 summarizes the basic logic functions for 1- and 2-input gates.

Table 6-1 – Summary of gate symbols and functions

Gate	Symbol	Truth table
AND		A B OUT 0 0 0 0 1 0 1 0 0 1 1 1
OR		A B OUT 0 0 0 0 1 1 1 0 1 1 1 1
NOT		A OUT 0 1 1 0
NAND		A B OUT 0 0 1 0 1 1 1 0 1 1 1 0
NOR		A B OUT 0 0 1 0 1 0 1 0 0 1 1 0
XOR		A B OUT 0 0 0 0 1 1 1 0 1 1 1 0

The cheapest 2-input gates to manufacture are the NAND and NOR gates. It can be shown that NAND gates alone (as well as NOR gates alone) can be used to reproduce all the other logic gates.

In-Lab Procedure

Follow the instructions below CAREFULLY.

□ Each critical step begins with a check box like the one at the left. When you complete a step, check the associated box. Follow the instructions below and carefully document your results for inclusion in your lab report.

6.1 Logic Gates

Logic Levels and Indicators

As a first step towards experimenting with digital electronics we will need a convenient source of logic levels to drive the gates with, and a method for monitoring the output states. Figure 6-1 shows a simple push-button circuit for generating logic levels and a simple LED circuit for monitoring a logic state.

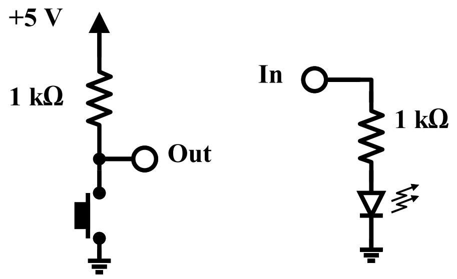


Figure 6-1 – Basic circuits for generating and monitoring logic levels.

□ Build two each of the circuits shown in Figure 6-1 on your breadboard. We will use a 5V supply throughout this lab.

We will improve the switch circuit a bit later in this lab. Note that simply connecting the switch output to the LED indicator circuit does not work well; these circuits work best when connected to CMOS gate inputs and outputs.

CMOS Logic

In earlier labs we constructed simple AND/OR gates using diodes, and built a CMOS inverter. More CMOS gates are shown in Figure 6-2. These can be implemented using the CD4007 chip that we experimented with in an earlier lab.

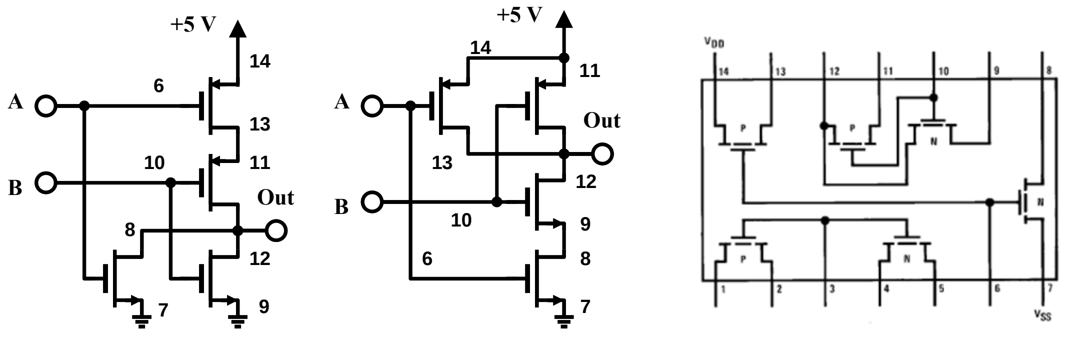


Figure 6-2 – Two CMOS gates that can be realized using the CD4007 chip.

□ Construct the circuits of Figure 6-2, and use two of your logic switches and an indicator to determine the truth table for each circuit. Record your observations. Remember that the switches shown in Figure 6-1 give a "low" level when pressed.

You should find that one of the gates implements a NAND function, and another implements a NOR function. These two gates are the easiest to build and can be used to implement all other logic functions.

SR Latch

While it is instructive to build the gates at the transistor level as we have just done, it would be cumbersome to do this every time we needed a logic function. Many CMOS chips are available that implement these or other combinations of logic functions. A popular family of CMOS combinational logic is the 4000-series. One example is shown in Figure 6-3, the CD4011, which includes four independent NAND gates.

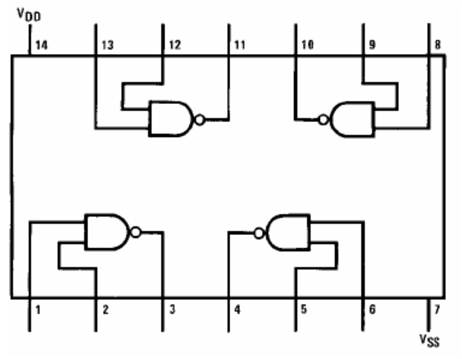


Figure 6-3 – Pinout of the CD4011 Quad 2-input NAND.

In other coursework you have probably begun to learn how to combine various logic functions to realize decision-making circuits. We will confine our attention here to the implementation of basic flip-flop structures, similar to the bistable multivibrator circuits discovered in the previous lab. Figure 6-4 shows an example of a set-reset or SR latch, implemented with NAND gates. This circuit is designed to make $Q=1$ whenever a "Set" event ($S=1$) takes place, and $Q=0$ whenever a "Reset" event ($R=1$) takes place. The circuit simply maintains or holds the current Q-state when the inputs are low ($S=R=0$).

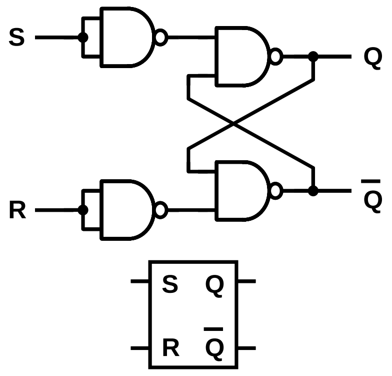


Figure 6-4 – Set-Reset latch implementation and symbol.

- ☐ Build the circuit of Figure 6-4 using the CD4011, two logic switches, and two logic indicators to monitor the outputs Q and $\overline{Q}$. Use a 5V supply. Construct a truth table to record your observations. Note that the $S=R=1$ state is unpredictable and not used. Don't forget to apply power and ground connections to pins 14 and 7, respectively.

The SR latch is essentially a static memory circuit. The circuit stores and maintains the output levels indefinitely until the next set or reset event.

D-flip-flop

Two simple modifications of the SR latch make it somewhat more useful, shown in Figure 6-5. Here a single input "D" is used in place of the "set" signal, and an inverter is used to generate the complementary "reset" input. These two input signals are gated by a separate "clock" signal "C", such that the outputs Q and $\overline{Q}$ will only change when "C" is

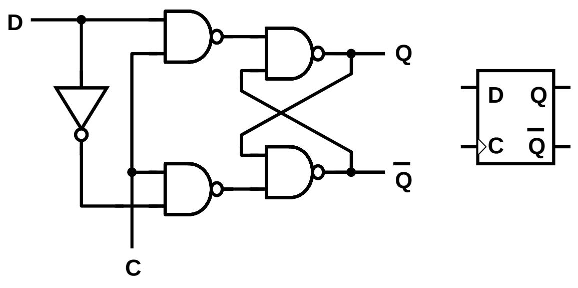


Figure 6-5 – D-latch and symbol

high. This "D"-latch thus avoids the ambiguity inherent in the SR latch when both inputs are high, since that state is no longer possible.

A further improvement to the "D"-latch is to add a second latch that is activated on the complementary clock signal, as shown in Figure 6-6. On the leading edge of the clock signal the first stage latches onto the input "D", and the output stage is deactivated. On the trailing edge of the clock signal the second "slave" stage is activated, latching on to the output of the first "master". Thus from an external standpoint, the output of the "master-slave" flip-flop appears to be triggered on the negative edge of the clock pulse. Because of the inherent advantages, most flip-flops are edge-triggered devices like this, although some may be *positive*-edge-triggered rather than *negative*-edge triggered like the one shown here. An example is the CD4013 dual D-flip-flop below.

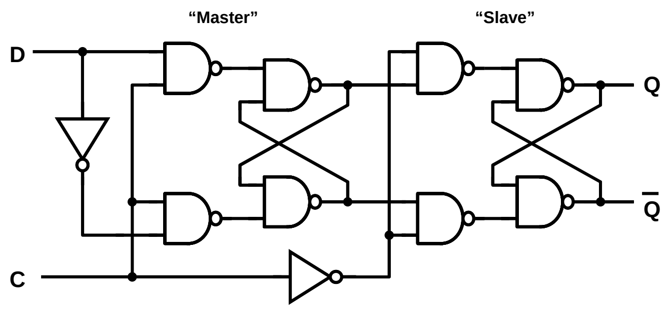
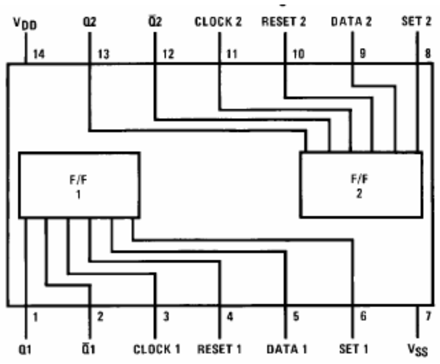
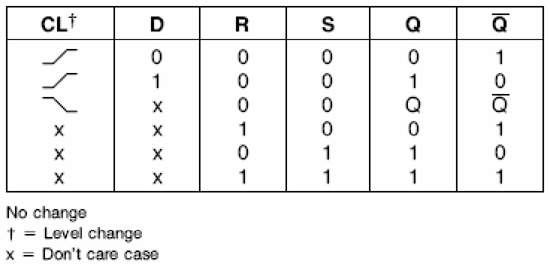


Figure 6-6 – Master-slave Edge-triggered D flip-flop





CL†	D	R	S	Q	$\overline{Q}$
↑	0	0	0	0	1
↑	1	0	0	1	0
↓	x	0	0	Q	$\overline{Q}$
x	x	1	0	0	1
x	x	0	1	1	0
x	x	1	1	1	1

No change
† = Level change
x = Don't care case

Figure 6-7 – CD4013 Dual D-flip-flop and test circuit.

The CD4013 chip shown in Figure 6-7 includes two independent D *positive*-edge-triggered flip-flops. In addition, each has separate set and reset functions that allow the outputs to be controlled independent of a clock signal, for added flexibility.

- □ Using the CD4013 and your logic switches/indicators, build the test circuit in Figure 6-8. When not used, the set and reset pins should be grounded. Don't forget to add power and ground connections to pins 14 and 7, respectively
- □ With a logic 0 applied at the clock input, vary the D-input signal; does the output Q change?
- □ Now apply a logic 1 to the clock input and vary the D-input level. Record your observations.

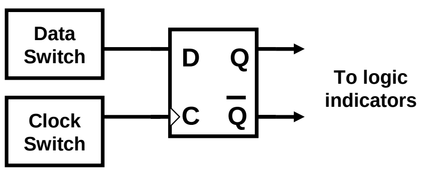


Figure 6-8 – Test circuit for the 4013.

6.2 Shift Registers and Counters

Switch Debouncing

Edge-triggered flip-flops require a nice clean signal transition. Unfortunately the behavior of most electrical switches is far from clean. A close examination of the switching transient shows that there is a short period of time where the switch output is erratic. This is caused by mechanical vibrations of the switch contacts, called "contact bounce", illustrated in Figure 6-9a. The duration may be on the order of hundreds of microseconds to tens of milliseconds depending on the type of switch. During this bounce period the output voltage can fluctuate wildly, leading to problems in an edge-triggered device like the D-flip-flop.

A popular solution to the problem is shown in Figure 6-9b. First, a capacitor is used to create an RC time constant that inhibits rapid voltage fluctuations. Secondly, a Schmitt trigger circuit (discussed in the previous lab) is used to create a sharp output transition and further avoid multiple triggering events. Here we are using a CD4093 device, which is functionally similar to the 4011 Quad NAND chip we used earlier, but in this case each NAND gate has been implemented with Schmitt trigger inputs. Note that with the NAND configured as an inverter, the output is now high when the switch is pressed. So this is an all-around improvement on the logic switch presented earlier.

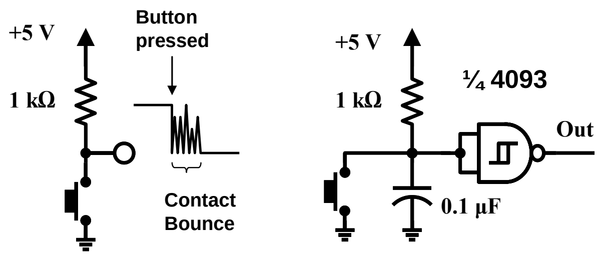


Figure 6-9 – (a) Switch bounce and (b) Solution.

- □ Build two more switches on your breadboard using the debouncing circuit in Figure 6-9b. Note that the pinout of the CD4093 is the same as the 4011 (see data sheet on the course website). When finished you should have a total of four logic switches.
- □ Also add two more logic indicators for a total of four.

Shift Register

One way to exploit the edge triggering is shown in Figure 6-10. Each time the clock signal is pulsed, the latches will acquire the signal from the previous stage. In this way, data is sequentially shifted from one latch to the next, moving from left to right. Such devices are used to translate a serial bit stream into a parallel output, among other uses.

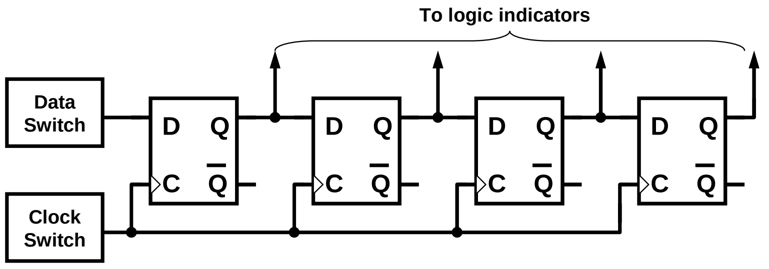


Figure 6-10 – 4-bit shift-register

- □ Using the two CD4013 devices in your kit, built the circuit in Figure 6-10.
- □ Place a logic-1 at the input and press the clock switch several times. What do you observe?

- □ Clear the outputs by pressing the clock switch several times with a logic-0 applied at the data input. Now, can you program the device to produce "1001" on the indicators?

Ring Counter and Johnson Counter

By looping the output back to the input, we can turn the shift register into a ring counter as shown in Figure 6-11. Here we have also added a provision to "preset" the state of the counter using the set/reset pins on the 4013. The choice of set or reset pin on each flip-flop will determine the initial state of the counter.

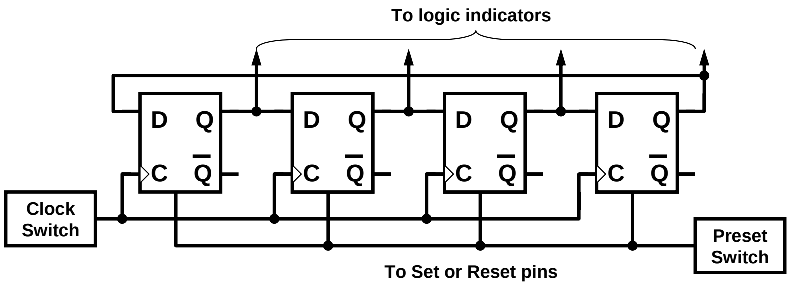


Figure 6-11 – Ring counter.

- □ Modify the shift-register circuit to construct the ring counter of Figure 6-11. Configure the "preset" so that the state "1000" is established whenever the preset switch is pressed. Now press the clock switch several times and record your observations.
- □ Configure your bench function generator to produce a 1Hz 0-5V square wave, and use this to drive the clock input in the ring counter.
- □ Now, modify the circuit so that the $\overline{Q}$ output on the last stage loops back to the input instead of the Q signal. This is a so-called Johnson counter or "twisted ring" counter. Preset the outputs to zero and drive the circuit again with the 1Hz clock. Record your observations.

Ripple Counter

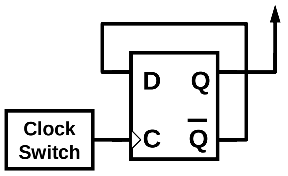


Figure 6-12 – Divide-by-2

Consider the circuit in Figure 6-12. This circuit is effectively a toggle switch. Each time the clock is edge triggered, the output is latched to its complement. This circuit also has the net effect of dividing the clock frequency by a factor of two, since a full period of the output signal requires two edge-triggering events at the input. By cascading a number of these divide-by-2 circuits we can create a full binary counter or divide-by-N network, as shown in Figure 6-13. Unlike the earlier circuits, each stage is clocked by the output of the previous stage.

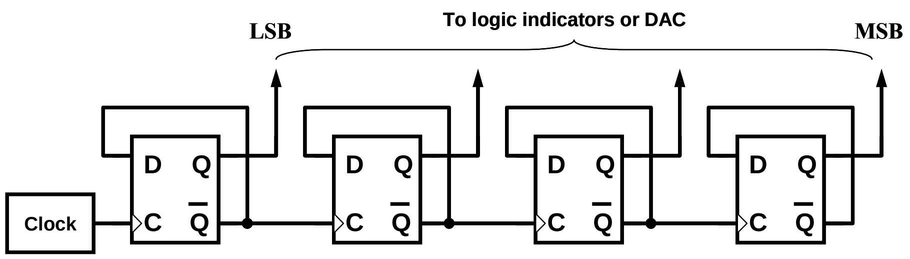


Figure 6-13 – Binary ripple counter (divide-by-16).

□ Modify your ring counter to build the ripple-counter of Figure 6-13

□ Drive the circuit with a 1Hz clock and record the behavior of the circuit.

□ Increase the frequency and observe the output of the last stage on the oscilloscope. How fast can the circuit operate properly as a 4-bit counter?

This kind of counter is called a ripple counter. Although it works reasonable well for small N, another type of counter—the synchronous counter—is usually preferred. In the synchronous counter all the stages are driven by the same clock signal so there is no delay in the updating of each bit.

6.3 Digital-Analog Conversion

As a last step we consider the problem of converting a binary digital word into an analog voltage.

In an N-bit system there are 2^N possible output states. We can associate each state with a voltage. If the maximum voltage in the system is $V_{\max}$, then the voltage increment will be $V_{\max}/2^N$, where N is the number of bits.

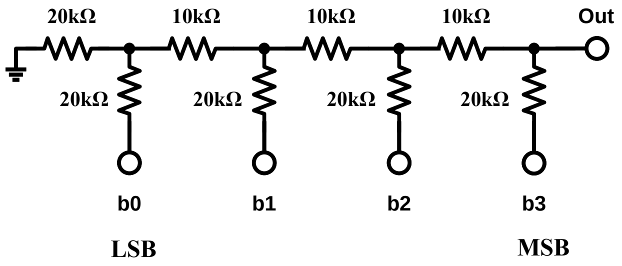


Figure 6-14 – 4-bit DAC using an R-2R network.

A common method for translating the digital number to an analog voltage uses the R-2R resistor network of Figure 6-14. It can be shown that the output voltage is given by

$$V_{out} = V_{\max}\left(\frac{b0}{16} + \frac{b1}{8} + \frac{b2}{4} + \frac{b3}{2}\right) \tag{6.1}$$

where $b0, b1, b2, b3$ are either 0 or 1. This is a simple but effective digital-to-analog converter (DAC). Many commercial DACs use this principle or a variant, with precision laser-trimmed resistors for accuracy.

If we use the ripple counter of the previous section to drive this circuit, we can generate all the possible output voltages in a staircase fashion.

□ Build the circuit in Figure 6-14 and hook it up to your ripple counter, with b0 as the least significant bit (LSB). You will need to remove all the logic indicators for the circuit to function properly. Drive the circuit with a 100Hz clock, and observe the output of the DAC on the oscilloscope. Record your observations.

Be sure you understand what is happening in this last circuit before leaving the lab.

Chapter 13 of Hands-on Electronics shows how a counter-DAC combination like this can be used in combination with a comparator to create an analog-to-digital converter (ADC).

Congratulations!

You have now completed Lab 6