Physics 323

Experiment # 10 - Digital Circuits

Purpose

This is a brief introduction to digital (logic) circuits using both combinational and sequential logic. The basic building blocks will be the Transistor – Transistor –Logic (TTL) NAND gate and the J-K Flip-Flop.

References

Storey *Electronics 4e*

(1) pp. 533-554 (Boolean logic and gates)
(2) pp. 591-602 (bistables)
(3) pp. 634-635, 643-647 (TTL devices)

Introduction

We will be using TTL (Transistor - Transistor - Logic) circuits where the two (positive) logic levels are: TRUE (or 1) = 5 V, FALSE (or 0) = 0 V. ***Warning:*** An unconnected pin or unconnected lead does not mean 0 V. It needs to be grounded if you want to insure that it is 0 V.

To examine the logical inputs and outputs of the various circuits we will be using a "logic setter" and a "logic detector" as shown in Fig. 1. The key to the logic setter is the SPDT (single-pole double-throw) switch. It is difficult to tell from the circuit diagram but the central lead of the SPDT switch is alternately connected to one of the two outside leads, controlled by the toggle switch. When connecting the LED you need to get the correct polarity. The flat edge of the LED is usually the cathode side. (Also the long lead is usually the anode but I have seen many exceptions among our LED's.) Use any colour that you like!

Connect the circuit of Fig. 1 to verify its operation. Make a mental note of which way the toggle is "flicked" for "ON" and "OFF".

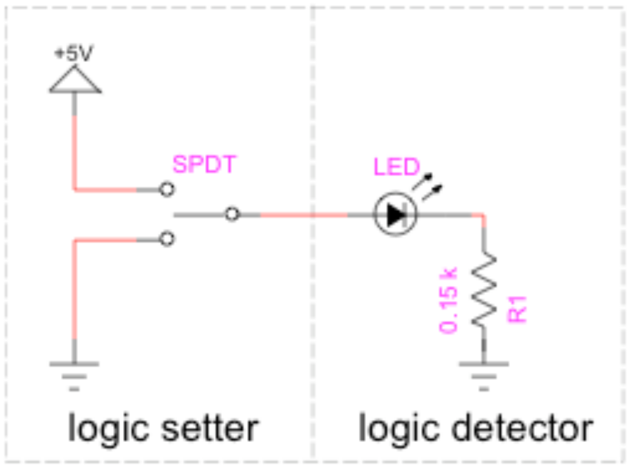


Figure 1: Using an SPDT switch to set a logic level and an LED-resistor combination as a logic detector

One of the most basic gates (and a *universal* gate) is the NAND which gives the function X = $\overline{\mathbf{A.B}}$ = /(A.B) = (A.B)' or, put another way, AND followed by NOT. There are variations with more inputs but we will just be considering the "2-input" gates here. ***Prelab:*** Look up the NAND truth table from your class notes or the text. In the TTL family the 74xx00 (where xx = LS in most of the chips and stands for "low power Schottky") is an integrated circuit that contains four separate NAND gates in a 14 - pin DIP (Dual in-line package) format i.e. it is a Quad 2-input NAND IC. You can find more information by looking in the *Texas Instruments TTL Logic Data Book* in the lab. Look up the device at the front of the book and then it points you to the correct page (2-3 in this case). We tend to be interested in the pin-configuration and the gates so it is usually just called a "7400" ignoring whatever "xx" happens to be (like most people don't know your middle name; mine is Philip!).

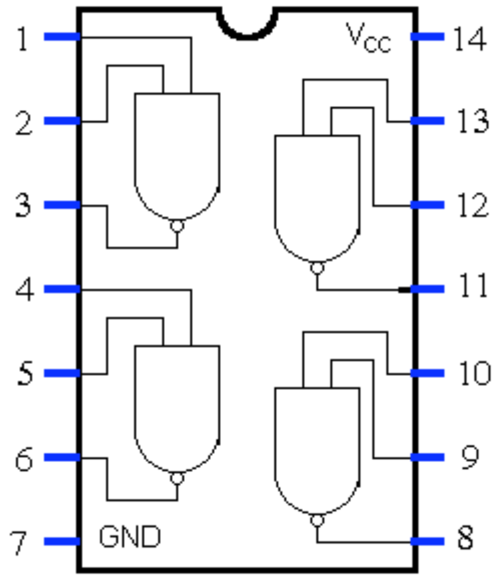


Figure 2: Pin configuration for 7400 Quadruple 2-Input NAND Gate IC

Prelab

(a) What is the purpose of the resistor in series with the LED? What is the current if you consider the LED to be an ideal Ga-As diode with a turn-on voltage of 2 V?

(b) Look up the XOR function truth table (and the NAND if you didn't already).

Lab-Combinational Logic

Connect up the 7400 using V_{CC} = 5 V (do <u>not</u> exceed 5.5 volts, and yes, the *CC* stands for collector). Connect one "logic setter" to pin 1 and another to pin 2. Connect a logic detector to pin 3. Verify the NAND truth table (i.e. the output state for each possible combination of the input states).

The other universal gate is the NOR gate i.e. all possible logic operations may be accomplished by only using NOR gates or NAND gates.

A less fundamental gate is the EXCLUSIVE - OR (XOR) which has the truth table. This could also be described as an "inequality" gate: the output is high when the inputs are not equal.

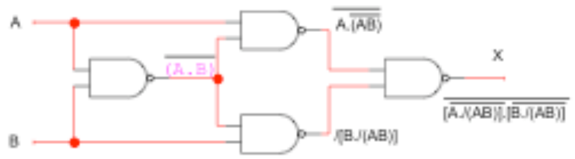

A
B
$\overline{(A.B)}$
$\overline{A.\overline{(AB)}}$
/[B./(AB)]
X
$\overline{[A./(AB)].[B./(AB)]}$

Figure 3: Implementing the XOR logic function with 4 NAND gates. There is a mix of overbar and / notation.

This logic function may be accomplished using 4 NAND gates. Build the circuit in Fig. 3 using the 4 NAND gates in the 7400 and check the truth table.

Lab-Sequential Logic

The circuits you have built so far are combinational circuits. You can generate many functions from such circuits - for example adders, multipliers, arithmetic and logic units (ALU's). etc - but none of these circuits have memory. They only respond to the input levels at the time; the outputs do not depend on their previous history. Such circuits which have 'memory' built into them are known as sequential circuits and the basic sequential circuits are latches and flip-flops. A very basic latch can be built from two NAND gates using feedback.

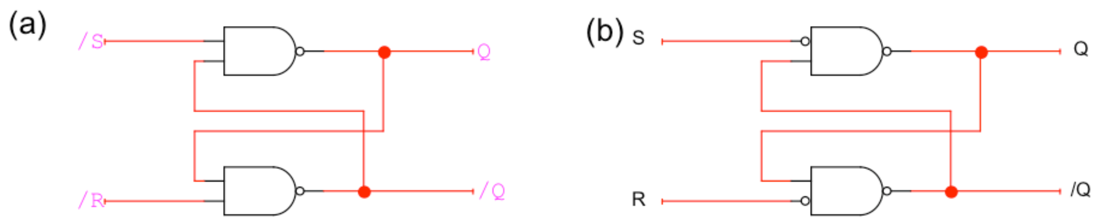
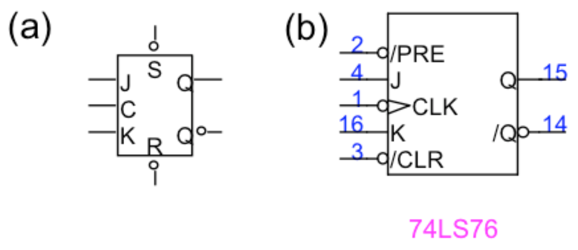


Figure 4: (a) The active low *SR* latch built from the 7400 NAND gates. (b) An alternate way of the depicting the active low *SR* latch with "bubbles".

This is a slight variation on the *SR* (Set-Reset) Latch. In the standard *SR*-Latch (made with NOR gates) the inputs are "active" when they are high but in this case setting /*S* or /*R* low "sets" and "resets" *Q*. The output labeled /*Q* *should* be equal to the complement or inverse of the *Q* output **except** if both inputs are low, then both outputs are also low. This situation is to be avoided in most cases and can lead to unstable outputs if you attempt to put the latch into memory mode (think back to what you know about positive feedback).

Build the circuit of Figure 4 using the gates in the 7400 and verify the latch behaviour. **Save this circuit!**

The more advanced (and useful) bistable circuits are activated by a separate clock signal to move them to the next iteration. As opposed to latches, flip-flops are activated on the edge of a clock-signal (*C* or *CLK* with a '>' for rising edge activation and '>' with a "bubble" for falling edge activation). One type of flip-flop is the J-K flip-flop - the 7476 contains two Master-Slave J-K flip-flops with clear, preset and complementary outputs. The "preset" function is active-low and thus acts like a "bubble" *S*. The "clear" function is also active-low and thus acts like a "bubble" *R*. On the chip diagram which you need to look up in the *TTL Logic Data Book* these are labeled as $\overline{PRE}$ or /PRE and $\overline{CLR}$ or /CLR



(c)

J	K	Q_{n+1}
0	0	Q_n
0	1	0
1	0	1
1	1	$\overline{Q}_n$

Figure 5: (a) Circuit symbol for *J-K* flip-flop with active low Set (Preset) and Reset (Clear). (b) A partial pin assignment for one of the *JK* flip-flops on the 7476. (c) The operation table. Remember the next "step" is determined by the falling clock edge.

Examine the operation of the *JK* flip-flop using the wave generator SYNC OUT as the clock input at a very low frequency ~0.1 Hz and with both /PRE and /CLR connected to +5 V (i.e. inactive). Momentarily ground each of /PRE and /CLR (with the other at +5 V) to examine their function. Do they need to be activated by the clock?

Connect the following circuit (with /PRE and /CLR inactive) and monitor the *Q*'s with LED's. -again use SYNC OUT at low frequencies as the CLK.

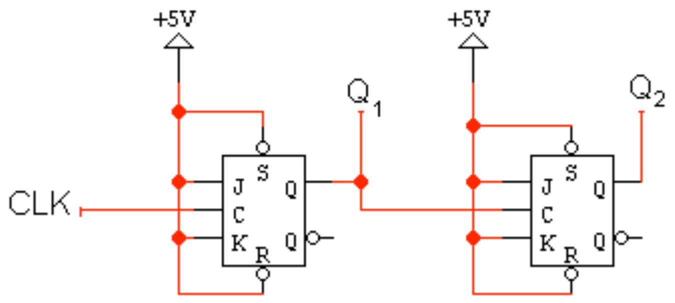


Figure 6: Two-bit binary counter.

To examine the behaviour at ~100 kHz measure each *Q* with the oscilloscope and compare with the CLK. Comment on the function of this circuit and **save it**.

With the circuit in Figure 6 we can test something we have covered /will cover in class: switch bounce and eliminating it with a latch circuit. Connect logic setter to the CLK input as it was connected to the LED in Figure 1. What *should* happen when you flick the switch? What does happen? Why? Does disconnecting the ground wire help or hurt?

We can eliminate the switch bounce with the following circuit.

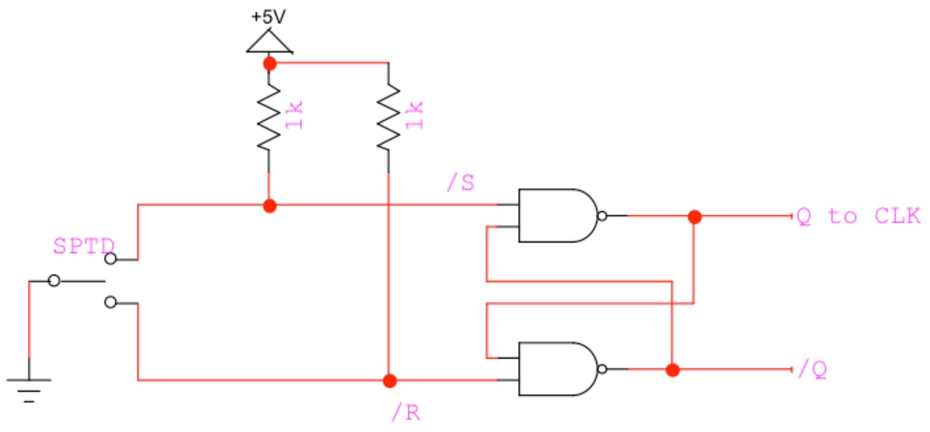


Figure 7: A switch "debouncer"

Construct this circuit using your two previously saved circuits and verify that it produces a predictable input to the counter.

CPA 2010-12-02