

## PHYS 331: Junior Physics Laboratory I

### Exercise on Digital Circuits

This laboratory exercise is designed to introduce you to the operation and use of some simple digital logic circuits. Very similar arrangements will be used later in various particle-counting experiments and in other data-acquisition applications.

Your write-up should explain what you did in each part, following the general organization of this manual. Include a short functional description of the circuit(s) built, a schematic of the circuit as you built it, and complete answers to all questions posed. A *neatly* hand-written report with clear free-hand figures is entirely acceptable.

Before starting the lab work you should review the topical notes on Digital Circuits from the course web site, and the pertinent sections of chapters 8 and 9 of *The Art of Electronics* by Horowitz and Hill.

#### A. Circuit Construction

Circuits are to be constructed on a breadboard using pre-cut jumper wires. We have also provided a digital "outboard" that plugs into the main breadboard with rows of pins. The outboard provides commonly used circuits for testing purposes: eight LED lights and two displays for monitoring states, eight logic switches for setting states, two push buttons for generating logic signals, and a square wave signal. It also connects the power supply to the breadboard through a regulator that reduces the voltage to +5 V. Documentation for the breadboard and the outboard is in the lab.

Most of the integrated circuits (ICs) we use come from the TTL family, identifiable by the 74xx numbering system that has become standard. The ICs are packaged in a 14-pin or 16-pin DIP (dual in-line package) which plugs into the breadboard. The pins on these devices are for inputs, outputs, and power connections, and there is a standard numbering system to identify pins on a device package. Data sheets for the TTL devices used in this lab, reproduced from the Texas Instruments TTL Data Book, are posted in the lab. Use these diagrams for determining pin numbers and as a quick reference for what the circuit does.

To insure proper operation of your circuits, and to avoid permanent damage to the ICs, follow these procedures:

- a. Don't insert or remove any device from the breadboard while power is applied.
- b. Use the IC tool when removing a DIP device from the breadboard. (It looks like a giant pair of tweezers.)

- c. All input leads on a gate that you are using must be connected to the output of a previous gate, to +5 V, or to ground. Open TTL inputs tend to go to logic 1, but this is not entirely reliable.
- d. Output leads that are unused may be left unconnected.
- e. Never exceed +5 V on the input of a TTL device, and be sure the ground (GND on the diagrams) and +5 V supply ( $V_{cc}$ ) connections are properly made. *Incorrect power connections are the easiest way to destroy any electronic device.*
- f. TTL outputs will usually drive up to ten TTL inputs. Violation of this rule causes no damage, but the circuit may not work.
- g. If your breadboard circuit appears defective, the problem is almost never a bad IC package, so exhaust every other possibility before replacing an IC.

Usually, more than one gate is in a single package. For example, four independent two-input NAND gates with common power connections are available in a single 14-pin DIP. To keep the total number of packages as low as possible, available gates are often connected to mimic the operation of other gates. For example, suppose we were building a circuit that required 3 NAND gates and 1 INVERT gate. Instead of using a package of four NAND gates with one left over, and a package of six INVERT gates with five left over, we could use the left over NAND gate to mimic an inverter, so that the circuit could be built with just one package of four NAND gates. Be sure to remember this technique when constructing your own circuits.

Schematic diagrams are intended to communicate the function of an electronic circuit to a knowledgeable "reader". Common sense is the best guide, but there are some conventions that you should take into account when drawing logic schematics:

- a. Organize your drawing so that the flow of information is from left to right.
- b. Use standard symbols for gates. Use a rectangle for more complicated devices, indicating function with appropriate labels, not just a pin number, on inputs and outputs.
- c. Inside or below the circuit element, give the device number. This is usually done only for complex devices such as counters.
- d. Above the line representing a connection to a terminal of a device (an input or output), give the pin number of that terminal. Pin numbers are often omitted for simple or non-critical devices.
- e. Ground and power connections to a device are not included in a schematic.

Note that a wiring diagram would be much more detailed, including all pin identifications, power connections and often some indication of the physical placement of components. For the simple circuits we deal with, actual wiring diagrams are neither needed nor really helpful.

The following sections describe circuits that can be built from gates. For each numbered section do the design of the system described and give a complete schematic for your design,

following the rules outlined above. Then construct the circuit and verify that it performs as claimed.

Before proceeding, a parenthetical comment is called for. If you were building a digital device that required, say, an adder or a multiplexer as a component in a larger system, do not build one out of individual gates as you will do here. There is almost certainly a single package which is cheaper, more reliable, and uses less power than a collection of discrete gates connected in your own, often idiosyncratic, way. So remember that in any real world application you should shop around for a device that performs your job with the fewest packages and the fewest discrete gates.

## B. Asynchronous logic

### 1. Verify a truth table

As a first, somewhat trivial, exercise in building a digital circuit, wire up a 7400 NAND gate and verify the truth table. The schematic for demonstrating the operation of the NAND gate is shown in Fig. 1. Note how the pin numbers for the 7400 are shown on the schematic. To make the wiring more explicit, all 14 physical pin connections to the 7400 device are also shown in Fig. 1. (You are not expected to document the physical layout of the circuits you construct.) When you have completed wiring the circuit, apply power to the breadboard and use the logic switches and LED monitor to see that the outputs agree with the truth table. Note that one input can control whether or not the output changes in response to the other input, hence the term "gate".

### 2. Binary addition

The rules for binary addition are given in truth table format in Fig. 2. A full-adder accepts as input two bits to be summed and a carry bit from a previous, less significant two-bit addition. The output is a sum bit and a carry bit. Full-adders can be cascaded to add a large number of bits.

Design a full-adder using the exclusive-or gate as one of the components of your circuit.

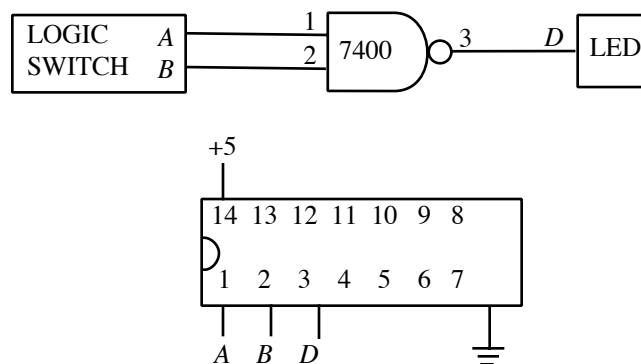


Fig. 1 Schematic and physical connections for NAND demonstration.

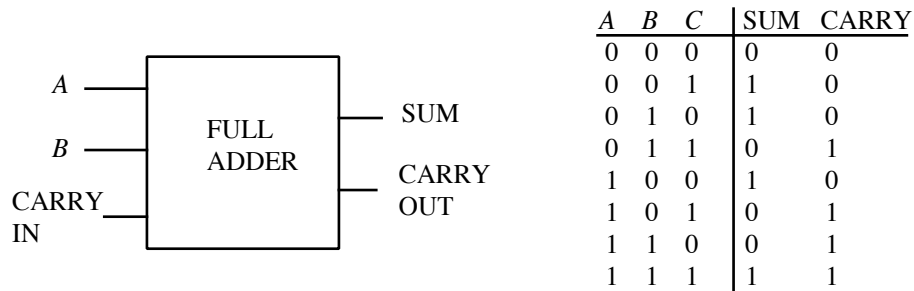


Fig. 2 Full-adder specification.

Convince yourself that your design will satisfy the truth table in Fig. 2.

Construct the full-adder, and connect the inputs A, B, and Carry-in to logic switches. Connect the Sum and Carry-out to LED monitors. Use the logic switches to verify that your circuit follows the logic rules of a full-adder.

### C. Clocked logic

#### 1. Flip-flops

The 74107 package contains two JK flip-flops which each have an additional "clear" input. Connect one of the flip-flops to the logic switches and to the outboard clock, following the pin diagram provided. Hold the clear input at +5 so that the circuit will operate, and verify the truth table. Also, use the scope to obtain a timing diagram showing the relation of the output at Q to the clock cycle. Does the output change as claimed?

#### 2. Binary counters

By connecting the Q output of one JK to the clock input of the other you can create a two-stage binary counter. Using the LED indicators, verify that the Q outputs count from 00 to 11 (binary).

To see the flip-flops function as a frequency divider, use the square wave from the outboard for the input clock. Observe the clock pulses and the output pulses  $Q_A, Q_B$  on the oscilloscope. What are their frequencies? Determine a timing diagram for the frequency divider, showing when the input and outputs change states relative to one another.

#### 3. Arbitrary counters

It is sometimes necessary to count pulses in groups that are not a power of 2. (For example, a photogate timer used for finding the period of a pendulum.) The 7490 decade counter package contains four master-slave flip-flops and additional gating to provide a divide-by-two counter and a three-stage binary counter for which the count cycle length is divide-by-five. Connecting the output  $Q_A$  from the divide-by-two to the input of the divide-by-five (INPUT B)

gives a counter that produces binary outputs corresponding to decimal 0 to 9. This is called binary coded decimal, or BCD.

Connect a 7490 so that it will count pulses from one of the logic pushbuttons, and display the results on 4 LEDs. Do you observe the expected sequence and a reset on the tenth pulse? When this works, connect the outputs  $Q_A...Q_D$  to the inputs A...D of the 7447 BCD to seven-segment decoder on the outboard. Your circuit should now display real numerals, and is ready to use in the next exercise.

## D. Data Conversion

The following exercises are intended only to demonstrate the principles of D/A and A/D conversion. Even very inexpensive commercial devices will have superior performance in all parameters.

### 1. Digital to Analog Conversion

Fig. 3 shows a simple scheme to convert a digital number from a counter chip to a proportional analog voltage. The op-amp is configured to sum the inputs with weightings in the ratio 1:1/2:1/4:1/8, corresponding to the relative binary values of the  $Q_A...Q_D$  outputs.

Construct the circuit as shown, using a breadboard with a digital outboard for the counter circuitry and another breadboard with an analog outboard for the op-amp. For initial testing, drive the 7490 counter with the outboard CLOCK so that it counts 0...10 (decimal) repeatedly. If you observe the op-amp output on the scope you should see a negative-going staircase, repeating after every 10 steps. Such a circuit might be used as a digital ramp generator, or as an arbitrary-function generator if driven with a more complex pulse sequence.

To further characterize the circuit, drive INPUT A from one of the outboard logic

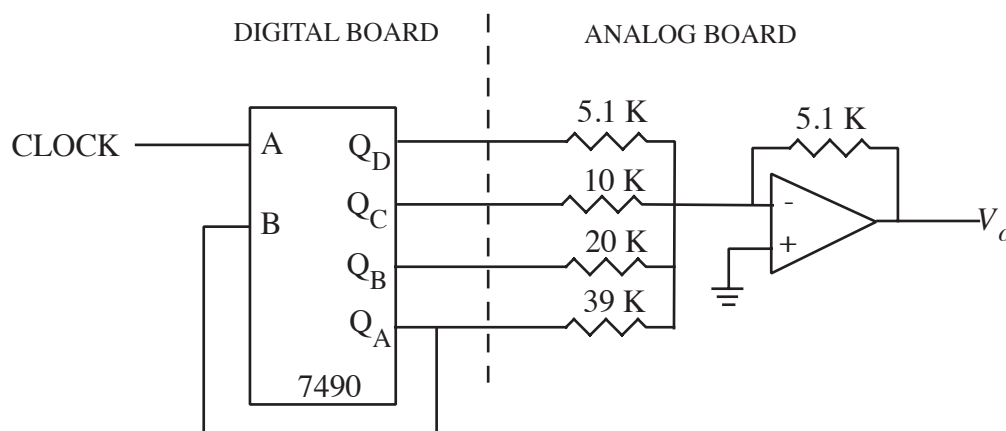


Fig. 3 A 4-bit digital to analog converter, with a 7490 counter to simulate digital input data. Supply voltage should be +5V for the 7490 and  $\pm 12V$  for the op-amp.

pushbuttons, so you can single-step the counter. Use a DMM to accurately measure the output voltage for each digital input, and make a plot of voltage vs digit. Is the graph linear? How big is the offset at zero input? What is the largest voltage deviation from the best-fit line? Offsets and deviations are usually expressed as fractions of the average step size (magnitude of the least significant bit), found from the slope.

## 2. Analog to Digital Conversion

A crude parallel or "flash" converter can be constructed from a set of comparators and a priority encoder, as suggested in Fig. 4. A comparator circuit has two inputs, one for signal and one for a reference voltage. The output circuit of the comparator chip chosen, an LM339, is a transistor switch which is "off" whenever the voltage at the signal input is greater than that at the reference input, and "on" otherwise. The 10 K $\Omega$  resistor connected to  $V_{CC}$  converts the switch state to logic one when the signal is greater than the reference, and logic zero when the signal is less than the reference.

The priority encoder, defined by the truth table in Fig. 5, produces a binary value equal to

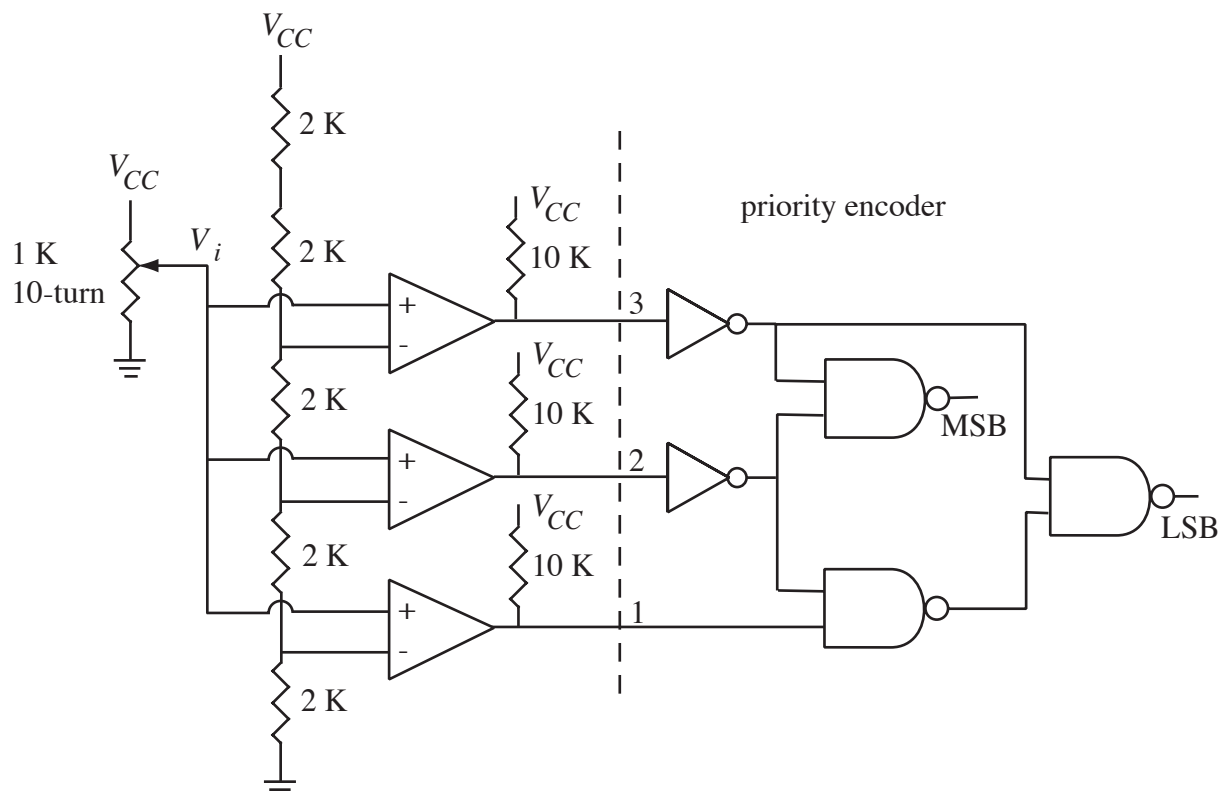


Fig. 4 A two-bit flash analog to digital converter using an LM339 comparator and TTL gates. The 1 K pot provides an input voltage for testing. Be very careful not to connect the variable resistor slider to  $V_{CC}$  or to ground, as damage will result. The supply voltage and  $V_{CC}$  are both +5V for TTL compatibility.

3	2	1	MSB	LSB
1	x	x	1	1
0	1	x	1	0
0	0	1	0	1
0	0	0	0	0

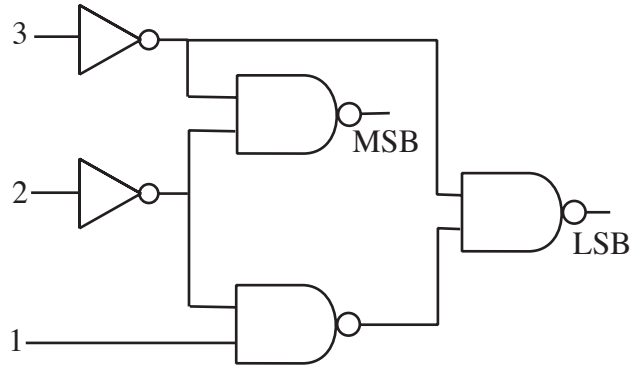


Fig. 5 Truth table and possible implementation of a 3-input priority encoder. Inputs are labeled 1, 2, 3 and output is a two-digit binary number, with the MSB being the two's place.

the number of the highest input that is at logic one. (The name arises because the same circuit is used to rank interrupts by priority in computer systems.) When connected to the comparators as shown in Fig. 4, the highest input number corresponds to the input voltage, completing the conversion.

Wire the circuits shown in Figs. 4 and 5 on a breadboard with a digital outboard. It is probably best to check the comparator and the logic portions separately before connecting the whole system.

With the subsystems connected, verify that the digital outputs properly follow the input voltage, using the LEDs for display. What happens near the transition from one binary output to the next? Can you explain why the LEDs seem to dim near the transition point? Hint: Look at the comparator output on the scope.

To complete your one-digit digital voltmeter you can connect the two outputs of the priority encoder to the A and B inputs of the numeric display system. (You will need to ground the C and D inputs, since they will go high and confuse the display if left floating.) What range of input voltage corresponds to each output digit?

### 3. Data transmission

As an example of data transmission between devices, we will use the circuit of Fig. 6 to eavesdrop on the commands sent by an IR remote control. This circuit will detect the IR signal from the remote control unit and produce a logic-pulse output for further processing. The sensor is a photodiode feeding a current to voltage conversion stage, followed by a low-pass filter. The comparator stage converts the received analog signal to a logic level. The amplitude of the analog pulses will vary as the transmitter moves but any IR output above the specified threshold will produce a full logic pulse from the comparator.

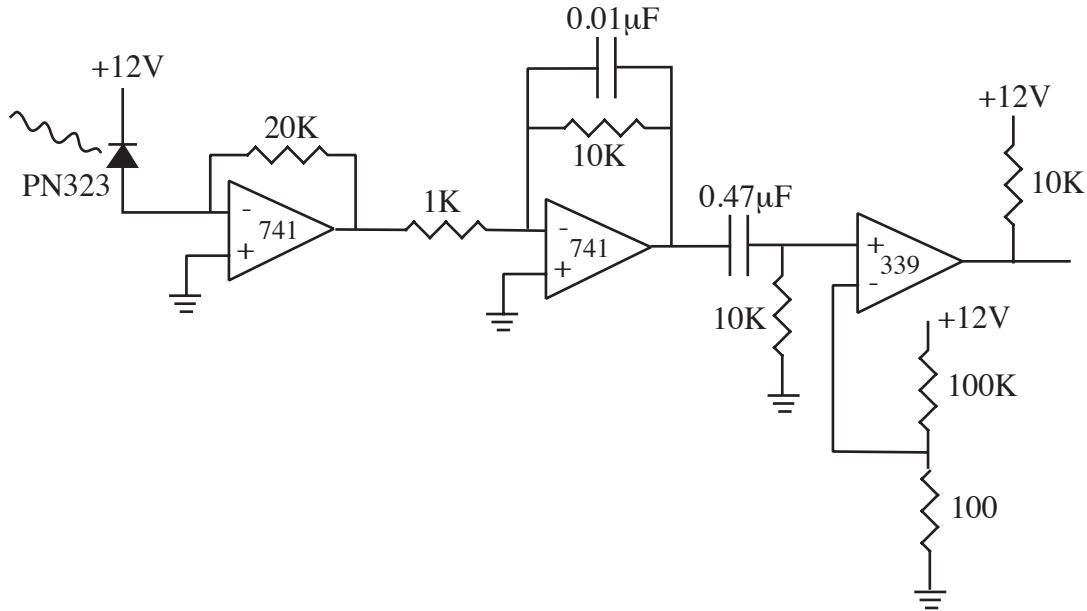


Fig. 6 A receiver for digitally-coded IR signals using 741 op amps and an LM339 comparator.

You should be able to see pulses at the output of the first op amp when you point one of the remote controls in the general direction of the photodiode. Careful use of the oscilloscope will show that the remote produces several bursts of light pulses, with individual bursts spaced by relatively long pauses. What is the repetition rate within the burst for the remote you are using? Typical values are tens of kilohertz. A commercial IR receiver would use a narrow-band filter passing only this carrier rate to reject extraneous noise and allow very high gain at the transmission frequency.

The low-pass filter circuit acts as an integrator, to smooth the bursts of carrier frequency into single longer pulses that represent the digital code. You can see these coding pulses at the output of the filter, along with the residual unfiltered carrier and background noise. Sketch a typical example of one code pulse at the filter output.

The smoothed pulses are AC coupled to the comparator input to remove any DC component due to ambient light or op amp imbalance. The comparator output consists of logic-level pulses when there is an IR pulse above the threshold set by the voltage divider. The time sequence of these pulses conveys successive digits of the command signal to the controlled device.

Once the circuit appears to operate correctly you can investigate various features. How far away can the remote be before you lose the signal? What is the analog signal to noise ratio going to the comparator when the IR signal is barely above threshold? Do you see any effect of ambient light at the op amp outputs? The remote will send out a burst of digital code representing the desired function when a button is pushed, and repeat the sequence at regular intervals as long as the button is held down. Can you deduce how numbers, which might be either decimal or binary, are represented in the pulse stream? Note: Seeing the command string on the oscilloscope is tricky. Try using normal

mode triggering and adjusting the trigger holdoff and variable time base to get a stable display of one repeat cycle. More venturesome students may try a digital storage scope, borrowed from the graduate lab across the hall.