Basic Digital Circuits

Due: By 6:00pm on Wednesday February 12.

1 Logic Gates and Circuits

In a digital system there are only two stable states, logic 1 and 0 (or HIGH and LOW, TRUE and FALSE, etc.) In a popular logic family called TTL (Transistor-Transistor Logic), the low logic level is assigned to 0V and the high logic level is assigned to 5V (see Section 3.10 of Wakerly's *Digital Design, Principles and Practices* for more information. Each logic gate is actually implemented in part of an integrated circuit (IC), with each gate made using several transistors. For the most part we do not need to concern ourselves with the actual circuitry inside each gate, only the interconnections between individual gates. Usually each IC package contains several individual gates. The 7408 chip implements 4 two input AND gates and is commonly referred to as a "Quad two input AND" chip. Similarly the 7432 chip is a "Quad 2 input OR" chip while the 7404 chip is a "Hex Inverter" since it contains 6 inverters. The IEEE standard logic symbols for each of these devices is shown in Figure 1. The IEEE symbols are better suited to represent ICs than the distinctive symbols since their rectangular outlines can easily fit together.

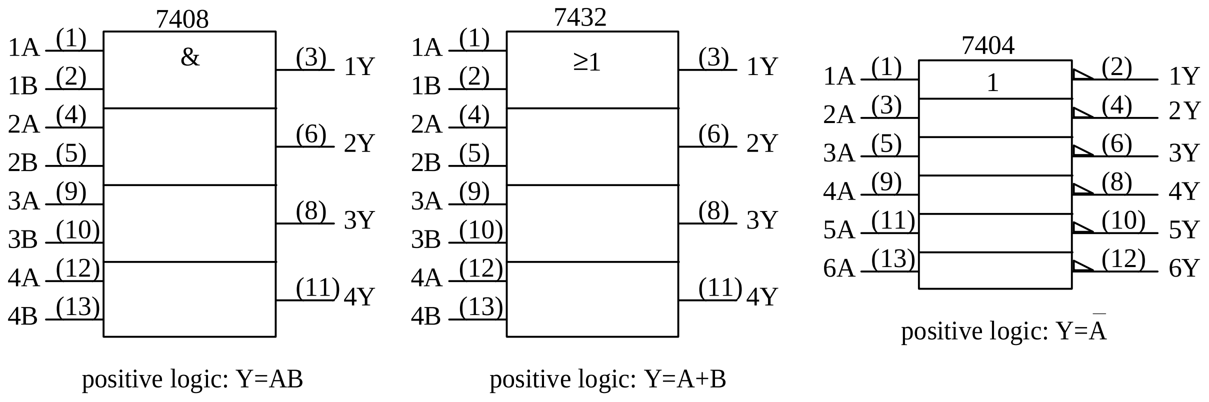


Figure 1: Standard Logic Symbols

2 Logic Circuits and Diagrams

Each of the chips in Figure 1 are available in 14 pin Dual-In-Line packages or DIPs. The pin numbers assigned to each logic signal are shown inside brackets in the figure. The pins are numbered as shown in Figure 2. Pin 1 is usually identified as the pin to the left of an indentation or cutout in one end of the chip that is visible when the chip is viewed from the top. Occasionally, it is also identified by a printed or indented dot placed just next to it.

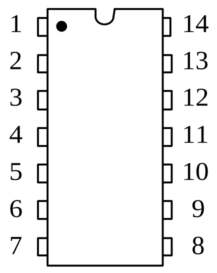


Figure 2: Identifying Pin 1

You will notice that pins 7 and 14 are not referenced in the logic diagrams. In 14 pin DIP packages, pin 7 is *usually* connected to ground (Gnd), and pin 14 is *usually*[1] connected to the 5V power supply (Vcc). These connections must be made or the chip will not work. TTL logic circuits are not passive! While TTL circuits are fairly forgiving, they can be destroyed by wiring mistakes. Take care not to connect pin 7 to power and pin 14 to ground, or to connect the outputs of two or more gates together. Only open collector and Tri-state gates are designed to operate with outputs of several gates connected together, and even in these cases, care must be taken.

A complete logic diagram including pin numbers using the 7408 and 7432 chips can be drawn as is done in either of Figure 3 or Figure 4,

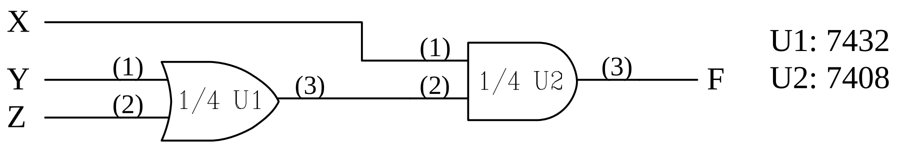


Figure 3: A Logic Diagram Using Distinctive Symbols

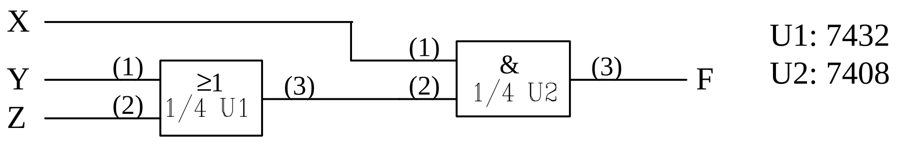


Figure 4: A Logic Diagram Using IEEE Standard Symbols

Notice that even though there are four AND gates on each IC, only the gates used are shown, and each gate is drawn separately. The U (unit) numbers can be used to identify which chip each gate is found on. A wiring diagram for this circuit is given in Figure 5.

The only diagrams you will need to make for this lab are the logic diagrams with part and pin numbers as in Figure 3 or Figure 4. They clearly convey how the circuit is intended to work logically, and they also give the chip and pin numbers which facilitate circuit implementation, testing and debugging.

[1]Occasionally, different pins are used, so be sure to check the chip's pinout information in a databook before making any connections.

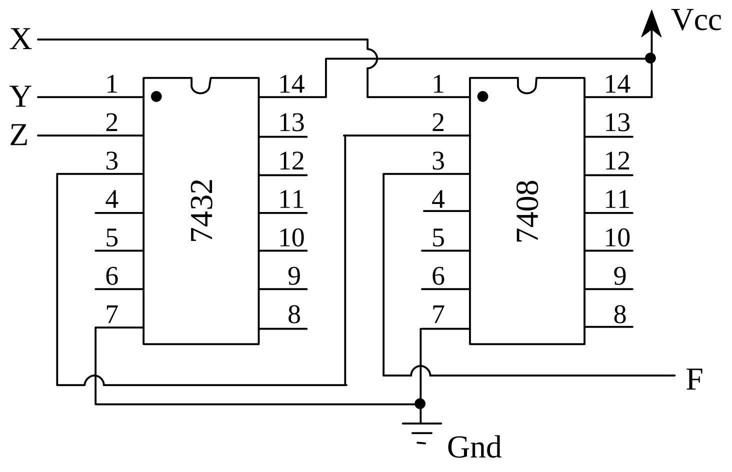


Figure 5: A Wiring Diagram

3 The Digilab Prototyping Board

Figure 6 shows how the circuit in Figure 5 can be implemented on a Digilab prototyping board. The circuit is constructed on the breadboard section of the Digilab board. A breadboard is used to rapidly create an experimental or prototype circuit without having to design and manufacture a costly Printed Circuit Board (PCB). A breadboard consists of an array of holes in which wires or component leads can easily be inserted. Rows of five or six holes are electronically connected to form a single node as shown in Figure 7. When a component lead is inserted into one of the holes, anything inserted into one of the remaining four holes will be connected to that lead. Nodes can be connected to each other using jumpers of .22 or .24 gauge wires with 1/4” of insulation stripped from both ends. The holes are spaced 100 mils (.1 inch) apart from each other, which is the standard spacing of the pins on a DIP package.

The breadboard has a groove down the center separating one side from the other. When inserting a chip into the breadboard make sure it straddles the central groove, otherwise the pins on opposite sides of the chip will be connected. Press the chip down until it touches the surface of the breadboard as shown in Figure 8.

Devices inserted on the breadboard can be connected to components on the Digilab board by using jumper wires between the device and the J2 connector. For example, switches SW1, SW2, and SW3 are used to input logic levels in Figure 6 and LD1 is used to indicate the output of the circuit.

4 Testing and Debugging

After a circuit is constructed it needs to be tested. This is usually done by observing the output as the inputs are switched through all possible combinations, and comparing the output with the desired output for each input combination. You will know a circuit doesn’t work properly when it gives the wrong output for a particular combination of inputs. The process of fixing problems in a logic circuit is called

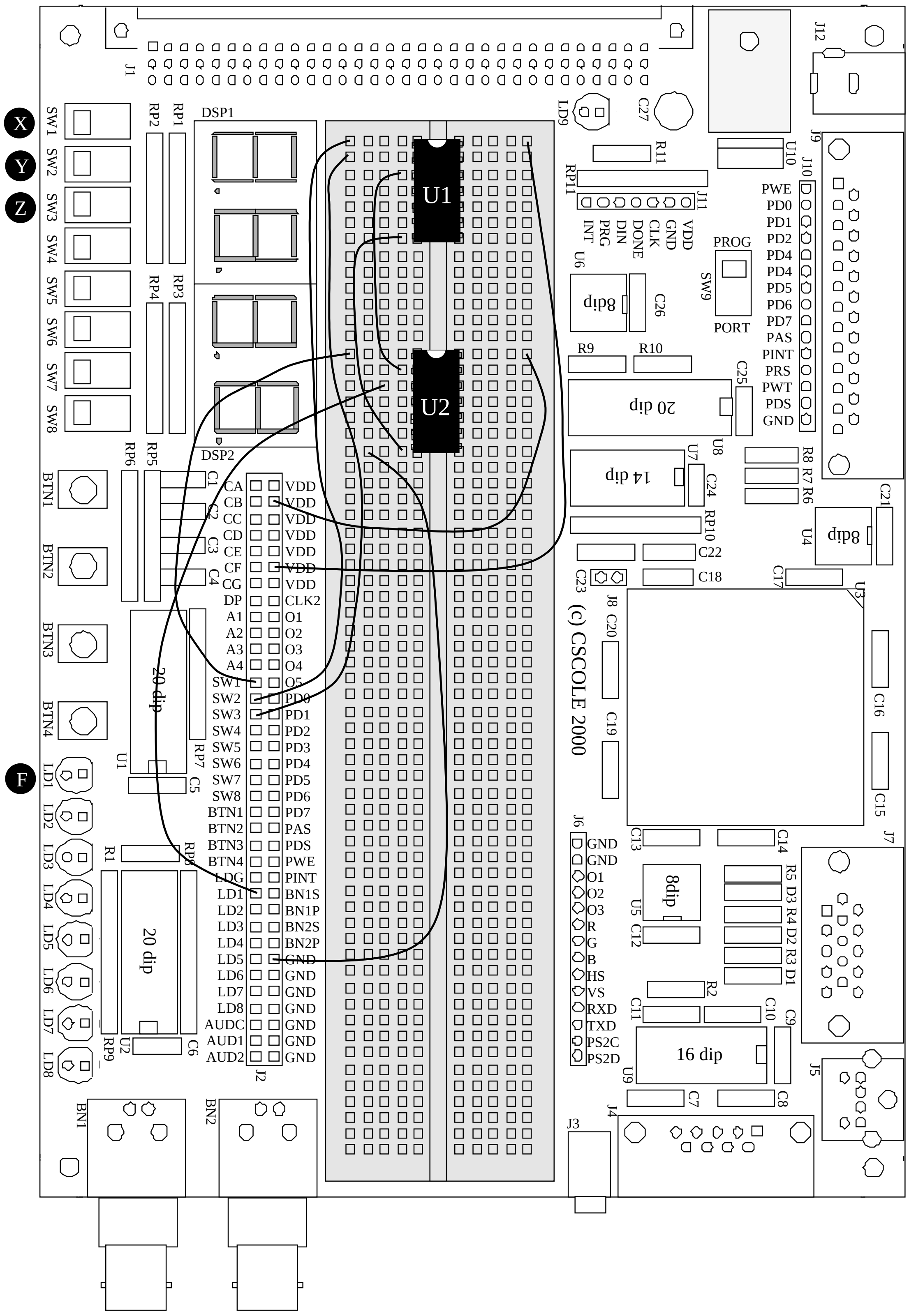


Figure 6: Implementation of a Circuit on a Digilab board

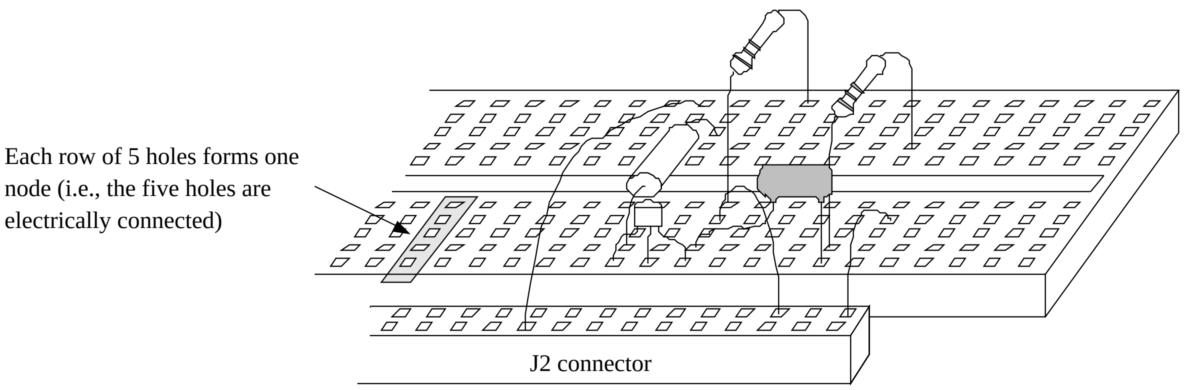
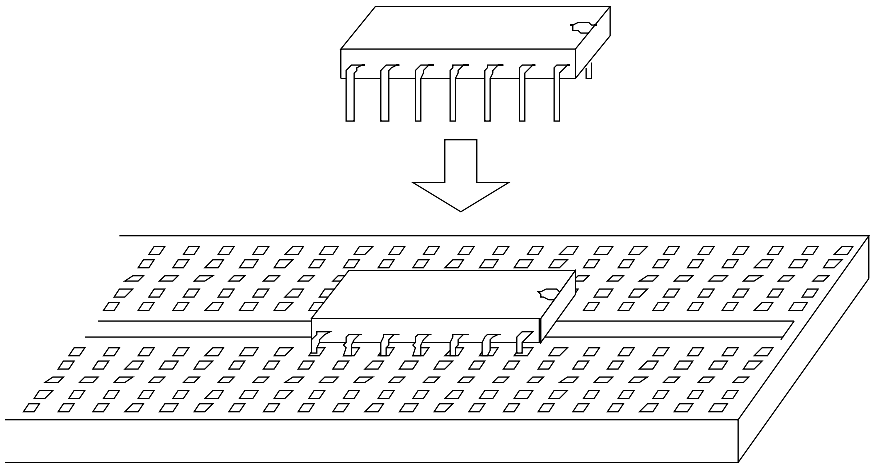

Each row of 5 holes forms one node (i.e., the five holes are electrically connected)
J2 connector

Figure 7: Making Connections on a Breadboard



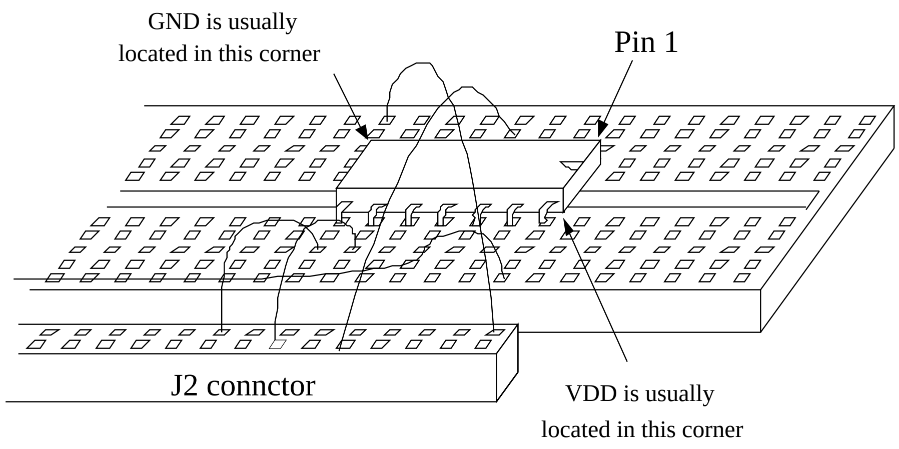

GND is usually located in this corner
Pin 1
J2 connctor
VDD is usually located in this corner

Figure 8: Placing DIP Devices on a Breadboard

"debugging".

When a circuit does not work, there are a number of possible problems.

1. Power and/or ground connections are not made to one or more chips in the circuit.
2. Two points that are supposed to be connected according to the logic diagram are not connected in the circuit. This can happen in a number of ways.
   (a) You forgot to make a connection when you were wiring the circuit.
   (b) You made the connection, but to the wrong pins.
   (c) A wire connecting two points has an internal break, so even though the points appear connected, in fact they are not.
3. You used different gates than the ones required, i.e.: you used a 7400 chip in place of a 7408 chip.
4. One or more gates in the circuit are "bad". A gate is "bad" when it does not implement the logic function for which it was designed.
5. Your design is faulty, meaning that your logic diagram does not implement the required truth table.

A number of these problems can be minimized by following careful procedures when designing and connecting the circuit. When designing a circuit, its a good idea to first make sure that the truth table or logic function you are attempting to implement is correct. Once you have a logic diagram, you should then check that it does indeed implement the required truth table. When entering part and pin numbers on logic diagrams, be careful that you are referring to the right part in the the TTL databook. The problem of forgetting to make a connection can be minimized by following a systematic procedure when constructing circuits such as checking off or red-lining connections (drawing over a connection in red pencil) on the logic diagram as they are made in the circuit. Make sure that you make connections to the right chip pins. Count carefully and make sure you know which pin is pin 1. Make sure that you put the right chips in your circuit. Do this by checking the designations printed on the chip, and by testing the chip in the chip tester. Make sure you know what unit number from you logic diagram belongs to each chip in your circuit. Put a piece of masking tape on each chip and mark the unit numbers on them if you have to.

Still, no matter how careful you are in the design and implementation of a logic circuit, problems will still exist. The logic probe is a useful tool for finding problems in digital logic circuits. [2] In the lab you will become familiar with its use. Following a systematic process to debug and correct a circuit is preferable (and often much less frustrating) than simply tearing out the circuit and starting again. The following steps are designed to help you fix most problems you will encounter.
When you find a problem . . .

[2]If a logic probe is not available you can "probe" a circuit using an LED in series with a resistor to find the logic level at a node. The LEDs on the Digilab boards are already connected to an array of resistors, so it is not necessary for you to provide a resistor between the node being probed and the J2 connector.

1. Use a logic probe to check that power and ground is properly connected to each chip.

2. Next, beginning at the output pin, move backwards though the circuit checking the logic at each point with the logic probe. For example, consider Figure 3. Assume that, for a particular combination of inputs, the output at pin 3 of U2 is supposed to be high, but is in fact low as indicated by an LED to which it is connected. First check pin 3 of U2 with the logic probe to make sure that it really is low and that the LED is not just giving a false indication (maybe it is burned out or the connection between pin 3 and the LED is broken). Assuming that the probe shows that pin 3 of U2 is low, we next check the gate inputs at pins 1 and 2 of U2. If pin 3 of U2 is supposed to be high, both pins 1 and 2 should also be high. If the probe shows that they are, then it *appears* that the gate is bad. However, the probe may show that one or both inputs are not connected to anything by indicating neither high nor low. Or it may show that one input is in fact low. In this case the next step is to check the gate output that is connected to this input. Continue this process, working back through the circuit, until a bad gate or a bad connection is identified.

3. When you find a gate that appears to be bad because its output does not agree with its inputs and the logic function it is supposed to implement, there are a number of things to consider.

   (a) The pins you thought were input pins may not be. Check the pin numbers for the inputs and output in the TTL databook. Also check that you properly identified pin one on the chip.

   (b) You may have the wrong chip. If you had used a 7400 instead of a 7408 for U2, the output at pin 3 should be low if both pins 1 and 2 are high. Check the chip number. The chip tester in the lab can also help you identify chips that have illegible or indecipherable numbers.

   (c) It may be that the output is being held at the incorrect voltage level by a bad connection. Consider what can happen if the outputs of two logic gates are accidentally connected together. One gate may be attempting to drive the output to a high voltage level while the other may be attempting drive the output low. One gate will be stronger than the other and effectively win this tug of war. To test for this situation, disconnect any wires that are directly connected to the output of the gate you suspect to be bad. Then test the end of each disconnected wire with the logic probe. If the probe shows a high or low value on any wire, then you have accidentally connected two outputs together and you should trace down and correct the offending connection. Since this situation can physically damage chips, you should probably test any chips whose gate outputs you have connected together.

   (d) The gate may in fact be bad. Turn the power off, remove the chip from the circuit and place it in the tester. If the chip tests bad (and even if it tests good) you should not simply replace it in your circuit. With the chip still out of the circuit, turn the power back on in the circuit and check the points in the circuit that were originally connected to the chip outputs. Any point that shows a high or low value indicates a point where two outputs were connected together. This situation needs to be corrected before the chip is placed back in the circuit.

4. If you do not find a bad gate or connection, then the problem is likely in your original design. Go back and check your work.

Once a problem has been corrected, you should retest the circuit by observing the output for all possible inputs.

5 Noise Margin

Earlier, we assumed that the high voltage in TTL is 5V and the low voltage is 0V. Figure 9 shows the steady state output of a 7404 inverter gate as a function of the input voltage at several operating temperatures. We notice that the actual range of output voltages appears to be from $0.2V$ to $3.5V$. We

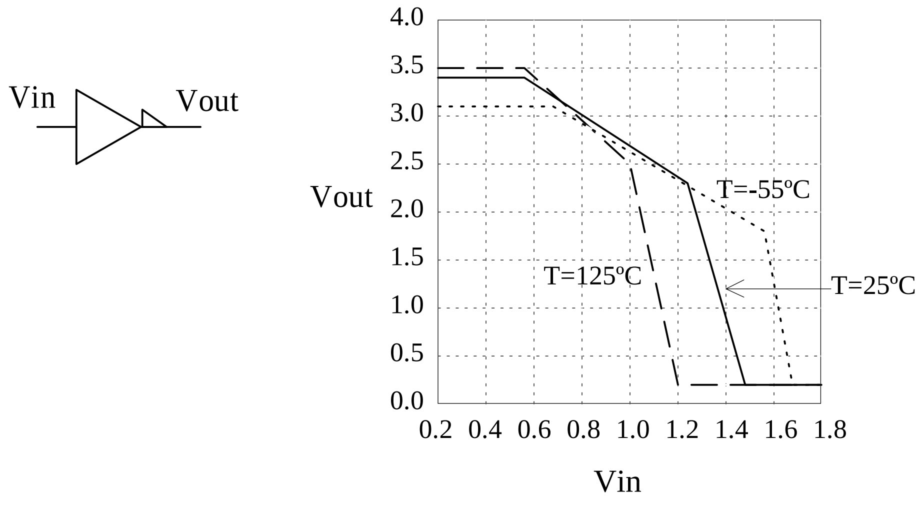


Figure 9: 7404 Inverter Transfer Function

notice that, independent of temperature, if $V_{in} > 2V$, V_{out} will be $0.2V$ so we can safely assume that the gate will interpret an input voltage greater than $2V$ as a high voltage. Similarly, Figure 9 shows that an input voltage less than $0.8V$ will reliably be seen as a low voltage. These two values are presented in Figure 10 as $V_{IH}min$ (V Input High min) and $V_{IL}max$ (V Input Low max). Each individual circuit will have slightly different characteristics, but the designers of TTL circuits guarantee that if the voltage on each gate input is either greater than $V_{IH}min$ or less than $V_{IL}max$, then the voltage on the gate output will never be less than $2.4V$ ($V_{OH}min$) for high output or never greater than $0.4V$ ($V_{OL}max$) for low output. Typically, output voltages will be nearer the values of $3.4V$ and $0.2V$.

Note that $V_{OH}min$ is greater than $V_{IH}min$. In other words, a TTL output produces a high voltage greater than that required by another TTL input. The difference between the two values, called the *noise margin*, is $0.4V$. The same margin exists for the low voltages. Thus, a line connecting a TTL gate output to a TTL gate input can tolerate up to $0.4V$ of noise added to or subtracted from the signal without causing the input gate to misinterpret the intended logic. Since noise is always present in electrical systems, it is important to have such noise margins.

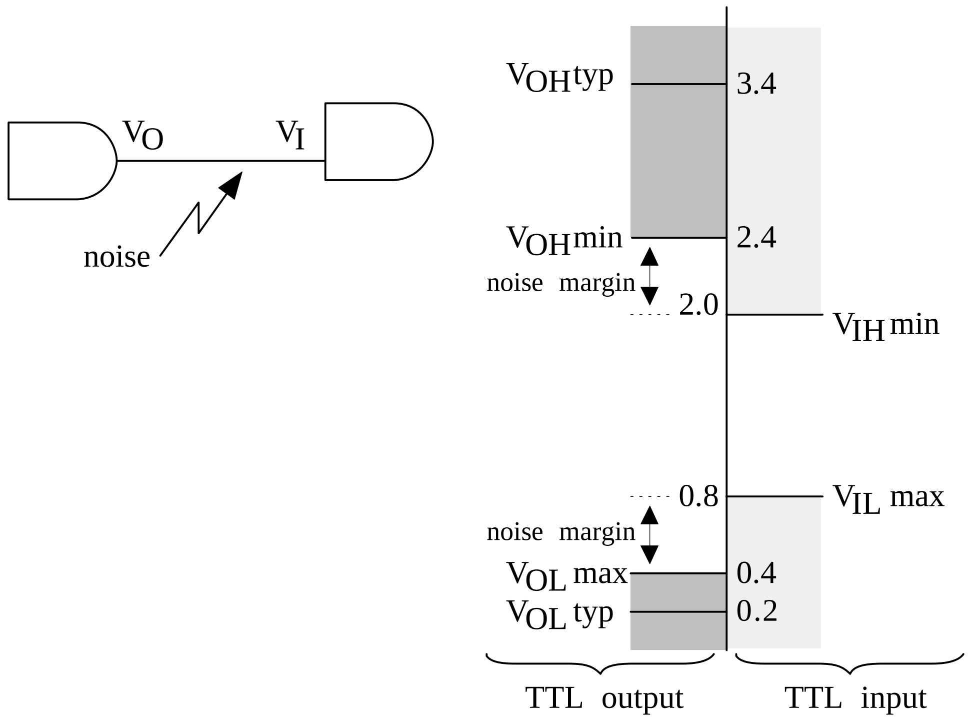


Figure 10: TTL Voltage Levels

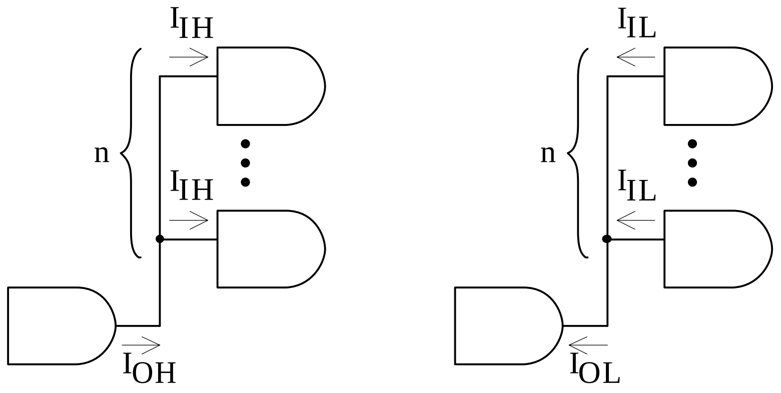
6 Fanout

Logic gates also have constraints on the number of individual gate inputs connected to each gate output. Each gate input connected to a given output represents an additional load that the output must overcome when it changes from high to low, or from low to high. Each gate output can source a current up to $I_{OH}max$ when its output is high. Each gate input can sink a current up to $I_{IH}max$ when its input is high. From Figure 11 and Kirchoff's current law, we can determine that the maximum number of inputs (n) an output can drive to a high voltage is given by the equation $I_{OH}max/I_{IH}max$. Similarly, we can determine the maximum number of inputs an output can drive low. The smaller of these two numbers is the maximum *fanout* of a device. For standard TTL logic gates, both these ratios are 20 so each TTL output should be connected to no more than 20 other inputs. [3]

7 Propagation Delay

Real logic gates also have *propagation delay*, that is there is a delay between a change in an input of a logic gate and a resulting change in the output of the gate. Figure 12 shows propagation delays for an Inverter and defines how propagation delay is measured. Since no voltage level can change instanta-

[3]In standard TTL, $I_{OH}max = 400\mu A$, $I_{IH}max = 20\mu A$, $I_{OL}max = 8mA$ and $I_{IL}max = 0.4mA$.



$$n = \min\left(\frac{I_{OH}\max}{I_{IH}\max}, \frac{I_{OL}\max}{I_{IL}\max} \right)$$

Figure 11: Fanout

neously, the input and output voltage waveforms in Figure 12 are shown with finite rise and fall times. t_{PLH} represents the low to high propagation delay and t_{PHL} represents the high to low propagation delay. The timing diagram in Figure 12 shows how the voltages correspond to logic levels. Rather than switch logic levels as the voltages drop below $0.8V$ and rise above $2.0V$, the switching takes place at $1.5V$ – halfway between the typical high and low values. This is not a bad approximation since rise and fall times are generally quite short.

Table 1 shows the propagation delays for the different styles of 04 type TTL inverters. The "S" and

	t_{PLH} (ns)		t_{PHL} (ns)	
	typ	max	typ	max
7404	12	22	8	15
74LS04	9	15	10	15
74S04	3	4.5	3	5

Table 1: Propagation Delays for 04 TTL Inverters

"LS" designations in Table 1 refer to "Schottky" and "Low Power Schottky" versions of TTL gates. These versions use different circuits to implement logic gates. The propagation delay of a gate increases with the fanout and can vary if different types of TTL gates are used together. The delays shown in Table 1 assume that the inverter output is driving only a single input of a similar type.

7.1 Timing Diagrams

The effect of propagation delay on circuits is analyzed by means of timing diagrams. Being able to construct them is a critical skill that anyone who wishes to design digital circuits must have. Figure 13

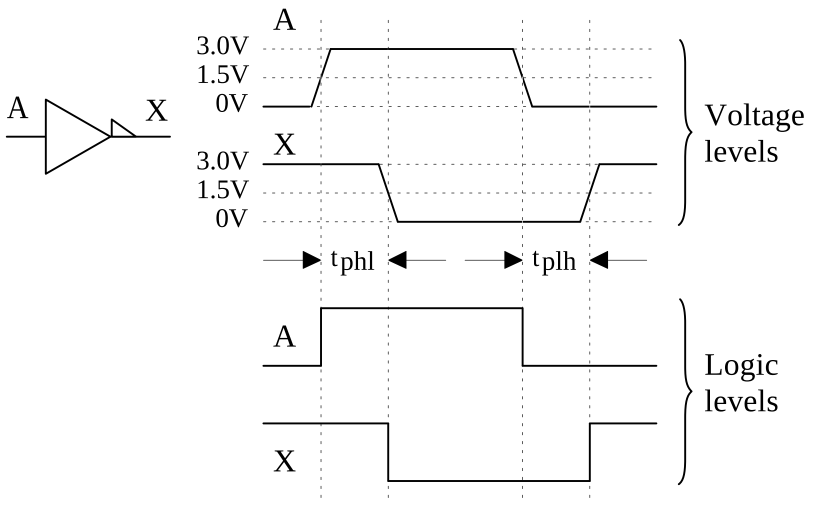
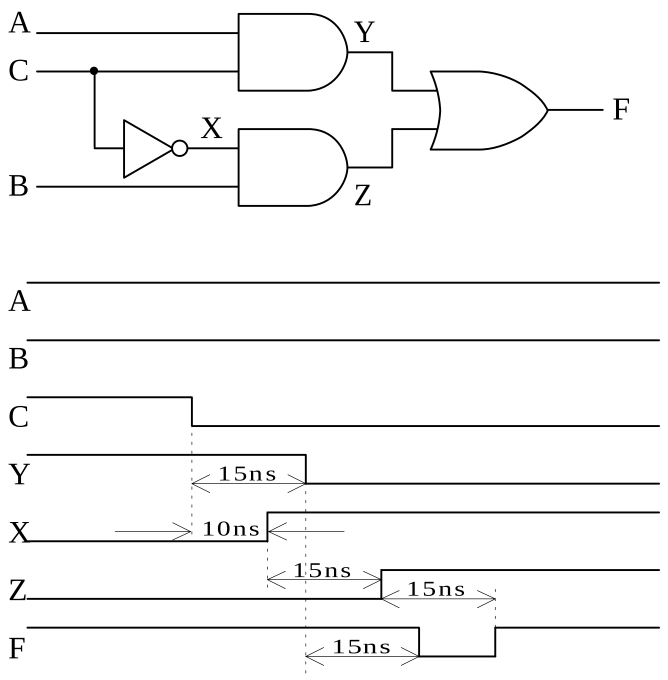


Figure 12: Inverter Propagation Delay

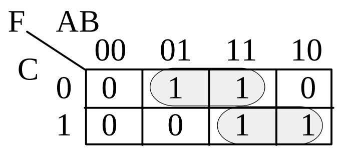
shows a truth table, a K-map (a tool for deriving a function from the truth table), a circuit implementing the function, and a timing diagram for the circuit. For this circuit we have assumed that all gates have equal low to high and high to low propagation delays – 10*ns* for the Inverter and 15*ns* for the AND and OR gates. When constructing a timing diagram we work from the inputs of a circuit to the outputs. In this example we consider what happens when the input C transitions from high to low. We will assume that A and B remain high the whole time. So we first draw lines for A and B showing them to be high and a line for C starting high and then, at some point in time, changing to low. We assume that A, B and C have been high for a long time prior to the point in time that the diagram begins so that we know what levels will be present at X, Y, Z and F at the beginning of the diagram. C changing from high to low will cause Y to change from high to low 15*ns* later. It will also cause X to change from low to high 10*ns* after C changes. With this knowledge we can complete the lines for X and Y. Next, the change in X will cause Z to change from low to high 15*ns* after the change in X. At this point we've considered all signals except F. Changes in Y and Z will both produce changes in F. At first you might find it difficult to consider when the output of a gate will change when both its inputs are changing. One way to do this is to draw a line for F assuming that the gate has no propagation delay and then shift the line to the right 15*ns*. Another way is to consider the first change first. Since Y changes before Z, we consider Y first. Since Z is low at the point in time Y changes from high to low, F will change from high to low 15*ns* later. Then F will change from low to high 15*ns* after Z changes from low to high.

7.2 Hazards

Notice that both before and after the change in C, the output of the circuit, F, is high, though the timing diagram shows that it does not stay constantly high. The temporary low in F is a result of propagation delays. Such an unintended result is called a *hazard*. In this case it is a *static one hazard* since the output was supposed to stay unchanged (*static*) at a high level. Whether or not this hazard is a problem depends



A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



$F = AC + B\overline{C}$

Figure 13: Timing Diagram Showing A Hazard

upon how the output is used. If it is simply used to drive an LED, or some other indicator then it may not be a problem since a 10*ns* off time will not be apparent to any human observer. On the other hand, if the output is connected to an electronic circuit, the unintended logic level may cause a problem depending on the details of the timing in the circuit to which it is connected.

7.3 Delay Models

Delays in physical systems can be classified as one or a combination of two basic types – pure delay and inertial delay. To explain the difference between the two types of delay, consider an electric motor. When a switch is closed to turn it on, the motor begins to turn, its speed increasing until it reaches its final speed. It can take several seconds for this process to complete. This delay is due largely to the inertia of the motor's rotor. One characteristic of inertial delay is illustrated by the fact that if the switch is turned off before the motor reaches its final speed, the motor immediately begins to slow down. This would not be the case in a pure delay system.

As an example of pure delay, assume that the motor is on a spacecraft headed for Jupiter. A command sent from the earth to turn the motor on could take tens of minutes to arrive. A second command to turn the motor off could be sent 1 minute after the first command was sent even though the first command has not yet arrived at the spacecraft. Both commands will ultimately arrive at the spacecraft, the command to turn the motor off arriving 1 minute after the one to turn it on causing the motor to run for a period of one minute. Timing diagrams for this example are shown in Figure 14. Note that the received signal is

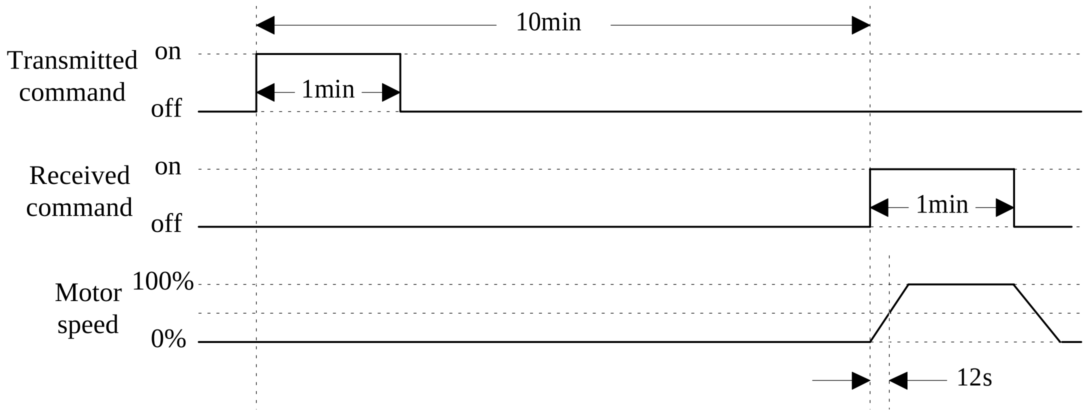
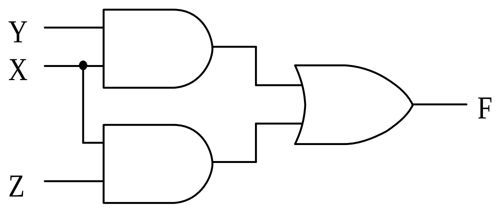


Figure 14: Delay Models

exactly the same as the transmitted signal except that it is shifted in time by 10 minutes. Pure delays can always be represented by simple time shifts of the signal. The signal itself is not modified. The inertial delay in the system is shown by the difference in the received signal and the motor speed. If we chose to say that the motor is up to speed when it has reached 50% of full speed, then Figure 14 shows that the system has 12 seconds of inertial delay. Thus, there is an overall delay of 10 minutes and 12 seconds between when the command to turn the motor on was sent, and when the motor is turning at greater than 50% of full speed.

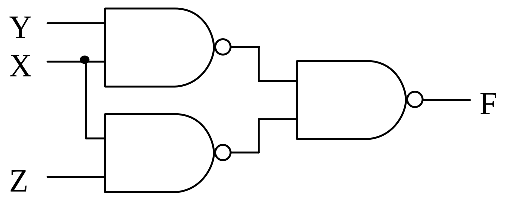
8 Preliminary Questions

1. In the preceding sections, all our examples involved gates and functions that had at most two inputs. Gates that have more than two inputs are very common. Consider the three input AND function $F = A \cdot B \cdot C$ (or $F = ABC$). In other words, F is 1 only when A is 1 and B is 1 and C is 1.
   (a) Find a TTL chip from the data sheets that is capable of directly implementing a 3 input AND function (that is $F = ABC$).
   (b) Draw a logic diagram showing how a 3 input AND function can be implemented using 2 two input AND gates and verify that your design is correct by finding its truth-table as was done in Figure 3.
2. Consider the following circuit:



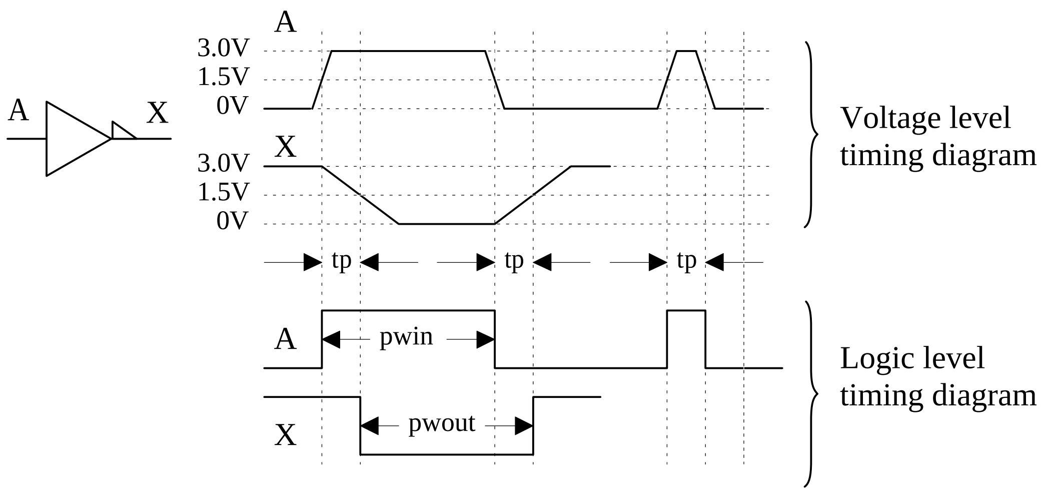
Obtain the Boolean expression and truth table that describes this circuit. Compare the truth table with the one given for the circuit in Figure 3. What can you say about these two circuits?

3. Obtain the Boolean expression and truth table that describes this circuit. Compare the truth table with the one you obtained for the circuit in question 2.



4. The propagation delay in a logic gate is largely inertial in nature due to capacitance in the circuit that prevents an output voltage from changing instantaneously.

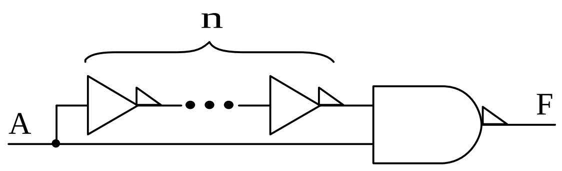
   (a) Complete the following voltage level and logic level timing diagrams for the output of a TTL inverter. Assume that the output of the gate only *begins* to change at the point in time that the input crosses the $1.5V$ threshold. Since it is an inverter, whenever the input rises above the logic threshold, the output should *begin* to fall. Whenever the input falls below the logic threshold, the output should *begin* to rise. The rate of output voltage change should be $1.5V/t_P$ where t_P represents the propagation delay of the device.



What happened to the output when the input was asserted (above the $1.5V$ threshold) for a period just equal to t_P when we use the above model? What would happen to the output if the input were to be asserted for a period of time less than t_P?

(b) Now, *assume* that the inverter is a pure delay device. That is, assume that it simply inverts the input logic and imposes a t_P pure delay (i.e.: pure time shift) on the output. We could model such a device by assuming it is constructed by connecting $t_P/10^{-9}$ feet of wire to the output of an inverter with *zero* propagation delay[4]. Add a line to the above logic level timing diagram showing what the logic output would look like if the inverter behaved as a pure delay device.

5. In the following circuit, n represents the number of inverters in the circuit. Using an inertial delay model for all gates, construct two logic timing diagrams for the circuit. In one assume $n = 1$. In the other assume $n = 3$. Assume that the inverters have $10ns$ propagation delays and that the NAND gate has a $12ns$ propagation delay. Start the timing diagrams with A high (assume that A has been high for a long time), then let A go low for $40ns$ and then high again.



9 The Lab

1. The LEDs (Light Emitting Diodes) on the Digilab board are used to visually indicate logic levels. Use jumper wires and the J2 connector to perform each of the following experiments and record your observations.

   (a) Connect one of the LEDs labeled LD1 through LD8 to VDD.

   (b) Connect an LED to GND.

   (c) Connect an LED to one of the switches labeled SW1 through SW8. Which way is High and which way is Low?

   (d) Connect an LED to one of the buttons labeled BTN1 through BTN4. Is the button High or Low when pressed?

2. Implement the circuits of problems 2 and 3 from the preliminary questions using 7400, 7404, 7408 and 7432 chips as needed. Draw a logic diagram for each circuit (as was done in Figures 3 or 4 including pin, chip, and unit numbers. Wire the circuits following your logic diagrams. Connect switches on the logic trainer to the circuit inputs and connect the circuit outputs to the LEDs on the Digilab board. Obtain truth tables for each circuit by stepping one at a time through all possible input combinations using the switches and recording the output displayed on the LED's for each.

   Demonstrate your working circuits to the TA.

[4]The pure propagation delay of an electrical signal in wire is about $1ns$ per foot.

3. Disconnect Vcc (+5V) from one of the chips in the above circuits. Measure the gate outputs on the chip that has Vcc disconnected. What do you observe? Reconnect Vcc and disconnect Gnd from the chip. Again probe the gate outputs and record your observations. Can this information help you test and debug logic circuits?

4. Wire up the circuit in Figure 15 as shown in Figure 16 where *n* is the number of inverter inputs connected to the output of a single inverter. The dashed lines represent jumper wires which can be connected or disconnected to vary *n*. Start with none of the dashed wires connected. Connect one

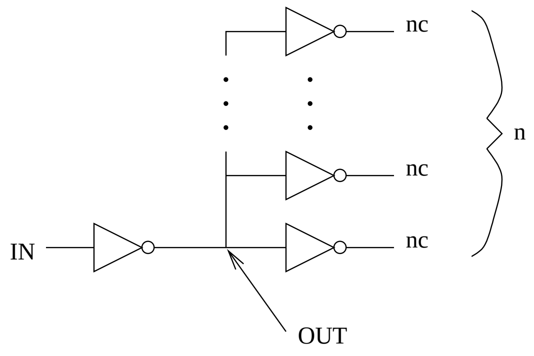


Figure 15: Schematic of Fanout Circuit

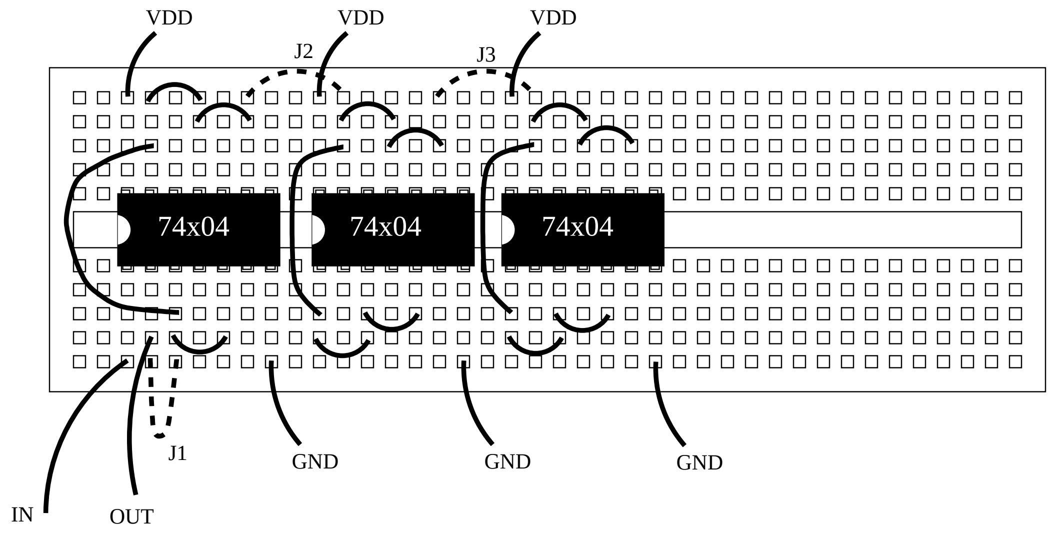


Figure 16: Fanout Experiment Setup

of the switches to IN. Use a multimeter to measure the voltage from GND to OUT. (To do this use “banana clips” to connect COMMON on the multimeter to GND on the Digilab board and V on the multimeter to your OUT signal on the Digilab board. Make sure that the ‘V’ switch is pressed on the multimeter to measure volts.) Record your observations and include them in your lab report. Record the voltages at OUT with both High and Low input signals for each configuration:

(a) No jumpers connected; zero fanout.

(b) J1 connected; fanout of 5.

(c) J1 and J2 connected; fanout of 11.

(d) J1, J2, and J3 connected; fanout of 17.

What can you conclude from your recorded voltages?

5. The following experiment will be set up by the TA. It does not require you to construct any of the circuits, but you should analyze and understand the circuits being demonstrated.

   Consider the following circuit, where *n* is an odd number of inverters connected one after another.

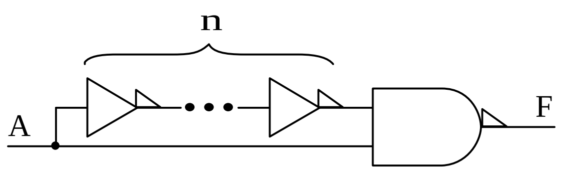


Figure 17: Schematic of Delay Time Experiment

(a) Complete the voltage level timing diagram in Figure 18 for $n = 1$, $n = 5$, and $n = 11$. You can use the 'X' line to calculate the signal after passing through the delay inverters, but before passing through the NAND gate. Assume all gates have the same propagation delay of $1.5V/tp$. If you need more intermediate steps you can draw the voltage diagram on engineering paper. Turn in your diagram to the TA.

Do not adjust anything on the oscilloscope.

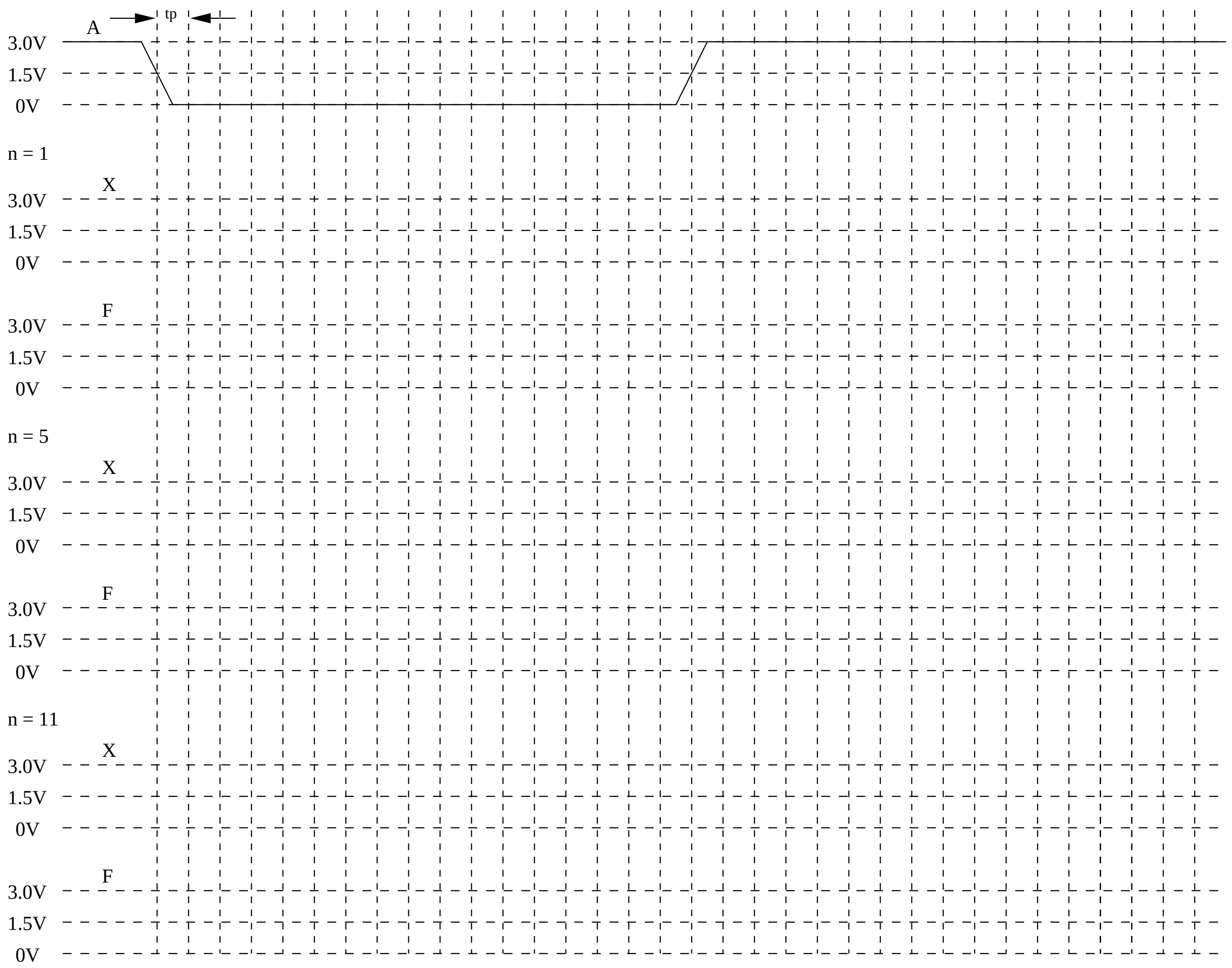


Figure 18: Voltage Level Timing Diagram for Delay Experiment