ELEC 2210 - EXPERIMENT 4
Binary Arithmetic Circuits

The objectives of this experiment:

In this lab you will review binary arithmetic, and you will learn to build digital logic circuits for the addition of binary numbers. The objectives of this experiment include:

- Learn to build and use binary arithmetic circuits
- Learn how to connect an LED to a CMOS output
- Continue to build experience with the ELVIS workstation and *Multisim* digital simulation
- Develop professional communication skills

I. Introduction

(Reference: Chapter 4 of Digital Logic Circuit Analysis & Design, by Nelson, et al)

One of the most fundamental uses of digital systems is numerical computation. Computers use the binary, or base 2, number system. Therefore we must learn how to perform binary arithmetic in order to build circuits to execute the desired operations of addition, subtraction, etc. Since the only possible outputs of a logic gate are 0 and 1, let's look at the possible arithmetic problems that arise from computing the sum of these two numbers:

Table 1. Base 10 Arithmetic Using 0 and 1

0	0	1	1
+0	**+1**	**+0**	**+1**
0	**1**	**1**	**2**

We can do the same thing in base 2 as shown in Table 2. In Table 1, the last result is written $1+1=2_{10}$, whereas in Table 2 it is written $1+1=10_2$ where the subscript is used to indicate the base being used.

The first three examples have results that can be represented as a digital output (0 or 1). However, in the last example, 1+1=2, the sum cannot be represented with a single binary value. (This corresponds to the base ten problem of 9+1=10; we can't write 10 as a single digit, so we "carry the one" into the tens place to get a two-digit answer.) In Figure 1, we show a detailed anatomy of the problem 1+1=2. In this view, we see that the binary result consists of a *sum bit* (0) and a *carry bit* (1).

Table 2. Base 2 Arithmetic Using 0 and 1

0	0	1	1
+0	+1	+0	+1
0	1	1	10

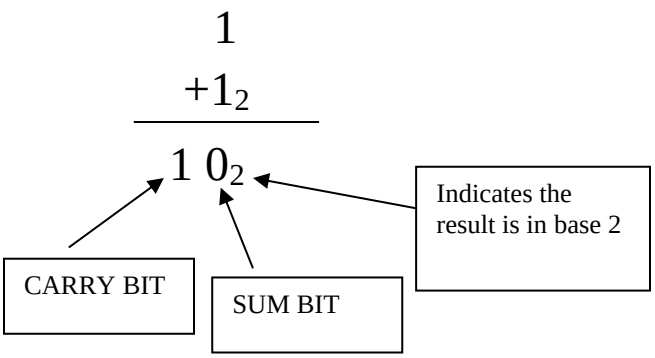


Figure 1. A detailed look at 1+1=2 in binary.

Half-adder: adding two bits

Table 3 is a truth table for the carry bit (C) and the sum bit (S) when two bits A and B are added.

Table 3. Truth table for adding two bits. The two input bits are A and B, and the two outputs are S (sum) and C (carry).

Inputs		**Outputs**	
A	**B**	**S**	**C**
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

The truth table shown in Table 3 can be implemented with the following Boolean logic:

$$S = A \oplus B \quad (1)$$

$$C = A \bullet B \quad (2)$$

The corresponding circuit is shown in Fig. 2. The symbol $\oplus$ stands for the logic function *exclusive-or (XOR)*, which is **one** if the inputs are different and **zero** if they are the same. The XOR is available as a standard IC logic gate (for example, the 74x86).

This circuit is called a *half-adder*. The reason for this name is that, although the circuit works fine for adding two bits, it has no provision for adding a carry-in bit, and therefore cannot be used directly in arithmetic involving more than two bits.

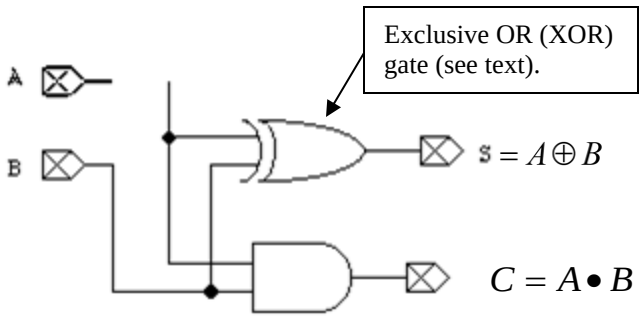


Figure 2. Half-adder logic circuit. The truth table is given in Table 3.

Full-adder: adding two bits plus a carry-in

The truth table for a two-bit full adder is shown in Table 4. The full adder accepts a carry-in, and can therefore be used for adding binary numbers with any number of bits simply by connecting together as many full adders as there are sum bits.

Table 4. Full adder truth table. The three input bits are the sum bits A and B, and the carry-in bit C_{IN}. The two outputs are the S (sum) and C_{OUT} (carry-out) bits.

Inputs				**Binary Outputs**	
A	**B**	**C_{IN}**	**Base 10 Sum of A+B+C_{IN}**	**C_{OUT}**	**S**
0	0	0	0	0	0
0	0	1	1	0	1
0	1	0	1	0	1
0	1	1	2	1	0
1	0	0	1	0	1
1	0	1	2	1	0
1	1	0	2	1	0
1	1	1	3	1	1

The full adder Boolean logic equations for the sum (S) and carry-out (C_{OUT}) bits are:

$$S = A \oplus B \oplus C_{IN} \tag{3}$$

$$C_{OUT} = A \bullet B + B \bullet C_{IN} + A \bullet C_{IN} \tag{4}$$

The corresponding logic circuit is shown in Fig. 3. The rules of Boolean algebra have been used to reduce the carry-out part of the circuit to the minimum number of logic gates, but it can be proven equivalent to Eq. 4.

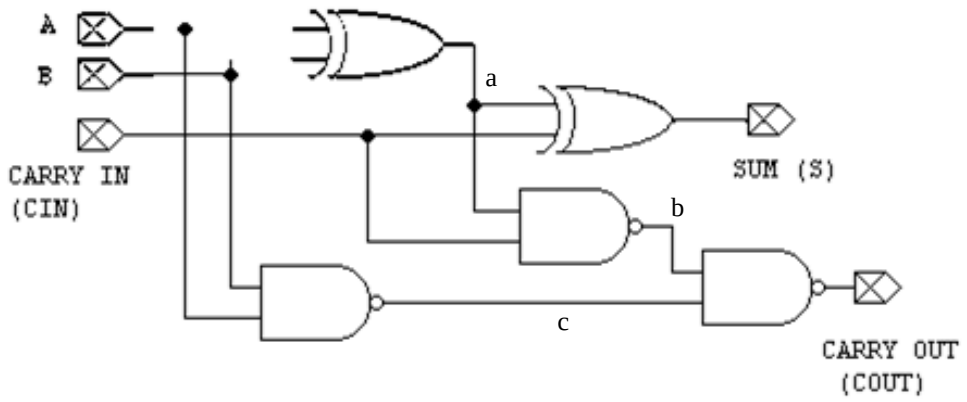


Figure 3. Full adder logic circuit. The truth table is given in Table 4.

The 74283: a 4-bit full adder IC

The 74283 (part number CD74ACT283E) is an IC that adds two 4-bit numbers plus a carry-in, and provides the four-bit sum and a carry-out. The pin-out diagram is shown in Fig. 4.

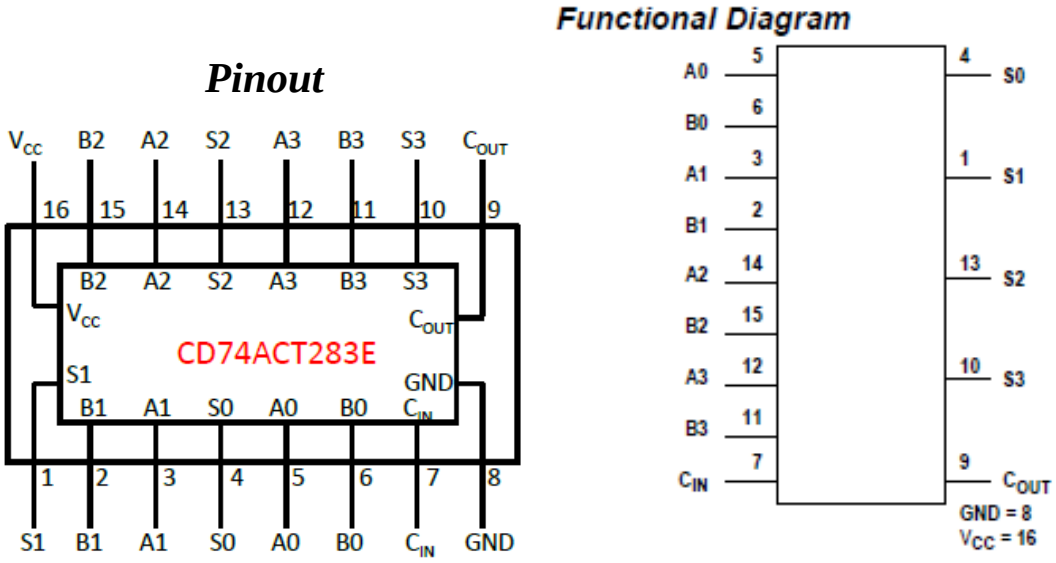


Figure 4. The 74283 (CD74ACT283E) four-bit full adder. The four bits of input A are labeled A3...A0, where bit A3 is the MSB. Input B is labeled similarly. The output sum bits are labeled S3...S0. The carry-in is labeled C_{IN} and the carry-out is labeled C_{OUT}.

Hexadecimal (Hex): A shorthand for binary

In the binary number system (base 2), the number of bits can be quite large. Present-day microprocessors use a 32-bit wide data bus. For writing logic functions and arithmetic expressions involving such large numbers, it is convenient to combine the binary digits (bits) into groups of four and represent them with base 16, or hexadecimal (hex) notation. Table 5 lists the decimal, binary, and hex notations for some numbers. The convenience of hex notation for larger numbers is clear from this listing. Note that each group of 4 binary digits is replaced by a single hex digit.

Decimal numbers are sometimes written with a following 'D', binary with a following 'B', and hexadecimal numbers are often written with a following 'H' to indicate the base. For example:

17 D = 0001 0001 B = 11 H

One other commonly used number system is base 8, called 'octal.' Although not discussed here, you are likely to run across it if you study computer systems.

Table 5. Decimal, binary, and hexadecimal notation.

Decimal (base 10)	Binary (base 2)	Hex (base 16)
0	0	0
1	1	1
2	10	2
3	11	3
4	100	4
5	101	5
6	110	6
7	111	7
8	1000	8
9	1001	9
10	1010	A
11	1011	B
12	1100	C
13	1101	D
14	1110	E
15	1111	F
16	1 0000	10
17	1 0001	11
32	10 0000	20
64	100 0000	40
128	1000 0000	80
1024 (1K)	100 0000 0000	400

II. Pre-Lab

1. Build and test a 1-bit full adder in *Multisim*.
   (a) Construct the circuit shown in Fig. 3 using INTERACTIVE_DIGITAL_CONSTANT parts for the inputs and Probe parts to display the Sum and Carry Out. Include the schematic in your Pre-lab.
   (b) Apply all 8 input combinations and record the outputs S and C_{OUT} in a truth table similar to Table 4. Are your results correct?

2. Obtain the data sheets for the CD74ACT283E 4-bit adder and the 74161 binary counter. These do not need to be turned in with the rest of the Pre-lab.

3. Use *Multisim* to build the 4-bit full adder circuit shown below in Figure 5. Include the schematic in your Pre-lab. Fill in a table of results similar to Table 6. Are your results correct?

4. The sheet attached to the end of this write-up contains the chip package diagrams of the adder and counter chips, plus the Digital Writer, 8 LEDs, and the ELVIS +5V and GROUND connections. On this sheet, using your *Multisim* schematic as a guide, create a wiring diagram for your experiment to use as a reference for wiring your circuit on the ELVIS workstation, i.e. draw all wires between component pins and devices.

III. Lab Exercise

Obtain the required components. You will use the following:

Qty	Part # or value	Description
1	CD74ACT283E	4-bit full adder, CMOS, 16-pin IC
1	74161	4-bit binary counter, TTL, 16-pin IC
1	LED	Red LED
1	220 ohm	Resistor

STEP 1. **Build and test the 4-bit adder.**

Use the CD74ACT283E to add two 4-bit binary numbers. As illustrated in Figure 5, one of the 4-bit numbers (we will designate it *A*) will come from a 74161 counter. The other (called *B*) is to be applied by the Digital Writer via bits DIO 3-0 and the Carry input (called C_{IN}) is to be applied by a the Digital Writer via bit DIO 4. Likewise, the 74161 clock is to be applied by the Digital Writer via bit DIO 7.

(a) On the ELVIS workstation, connect the circuit according to the pinout from your Pre-lab. Use four ELVIS LEDs to show the value of input A (from the counter outputs), and use the other four LEDs to show the sum S.

(b) Connect a separate LED/resistor combination to display the CD74ACT283E Carry-Out (C_{OUT}). In order to limit the LED current and to protect the chip as well as the LED, it is necessary to put a resistor in series with the LED. For a standard CMOS output, it is desirable to limit the current to approximately 24 mA. For a standard red LED, this can be accomplished by using a 220 Ω resistor as shown in Figure 6.
   1. Obtain a resistor marked 220 ohms (red-red-brown). Measure the actual resistance with the digital multimeter or DMM, and record the value.
   2. Obtain a red LED and connect it and the resistor to the carry output pin of the adder chip, as shown in Figure 6. The LED terminals are shown in Figure 7.

(c) Test the circuit by filling in a table like the one below (Table 6). Use the exact inputs given. Verify that your LED comes on when the carry output is high, and turns off when the carry output is low. (If not, perhaps your LED is in backwards – refer to Figure 7.) Put this table in your report. Have your instructor observe your circuit in operation and initial the table in your report.

STEP 2. **Measure LED and CMOS output characteristics.**

(a) Using the DMM, measure the voltage across the resistor when the LED is on. Using Ohm's law, calculate the LED current. Compare your calculation with the desired maximum value of approximately 24mA stated earlier.

(b) Using the DMM, measure the output high and output low voltages on pin S0 (connected to an ELVIS LED) and pin C_{OUT} (connected to the LED/resistor combination). Record these output voltages first with the LED connected to the pin being measured, and again with the LED disconnected from the pin being measured. ***(Make sure you disconnect the LED from the pin being measured – ELVIS LED when measuring pin S0, and the red LED when measuring***

pin C_{OUT}.) Record the measured values in Table 7, and put this table in your report. Have your instructor observe your circuit in operation and initial the table in your report.

STEP 3. Cleanup.

DO NOT PUT CHIPS OR ANY OTHER COMPONENTS IN THE WIRE TUBS.

(a) Turn off the power to the ELVIS workstation.
(b) Disassemble your circuit and place all wires back in the wire tub.
(c) Put all chips and other components back in the proper bins.
(d) Clean up your work area and discard any trash.

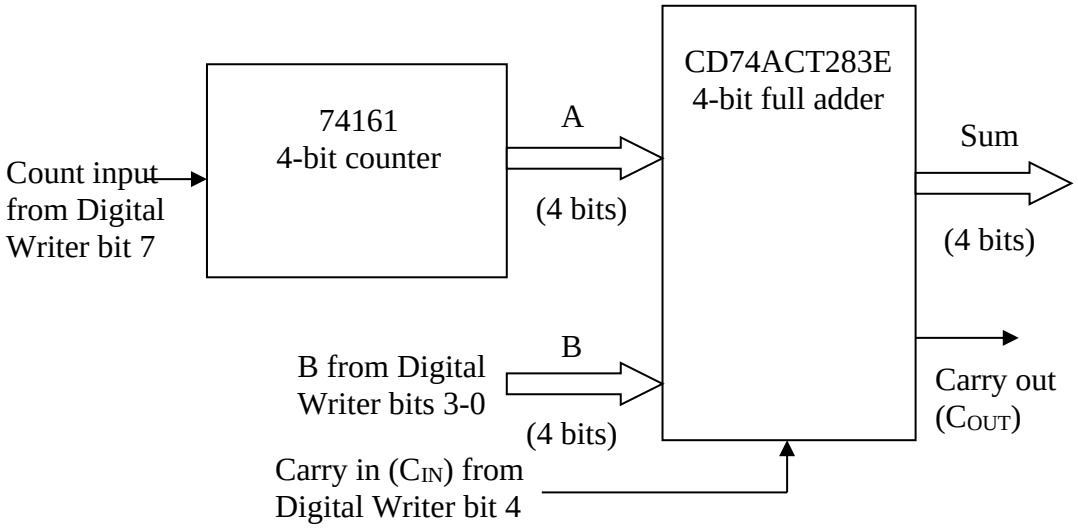


Figure 5. Block diagram of 4-bit full adder circuit.

Table 6. Test results for the 4-bit full adder circuit. All values in hex.

Inputs			Calculated Outputs		Actual Outputs		
A	**B**	**C_{IN}**	**C_{OUT}**	**S**	**C_{OUT}**	**S**	**Check here if correct.**
0	0	0					
0	0	1					
1	0	0					
7	7	1					
8	7	0					
8	8	0					
F	F	0					
F	F	1					

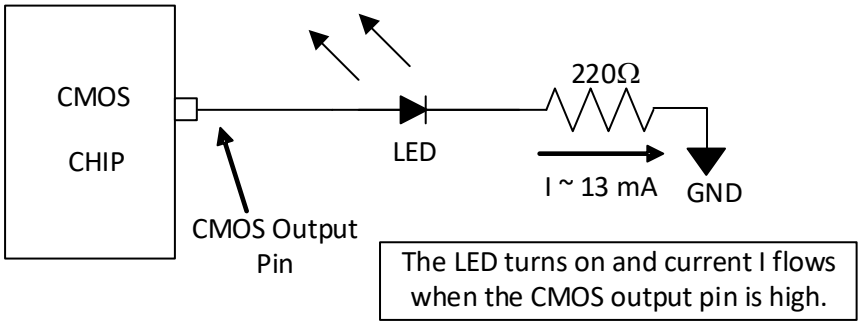


Figure 6. Connecting an LED to a CMOS output pin.

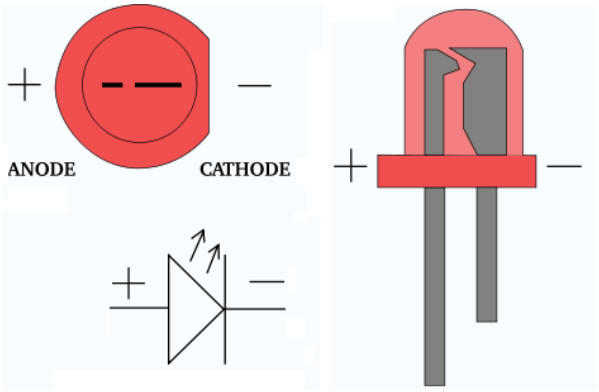
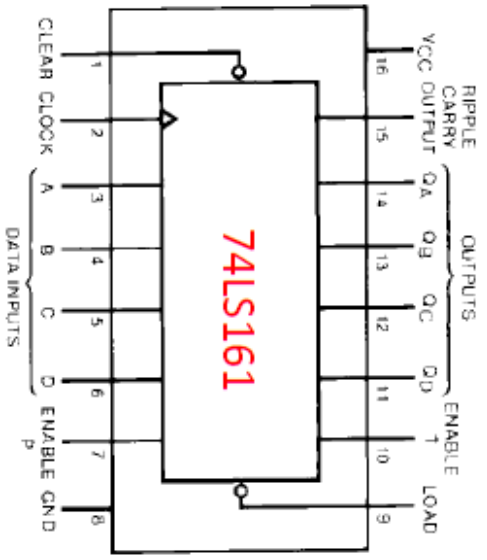
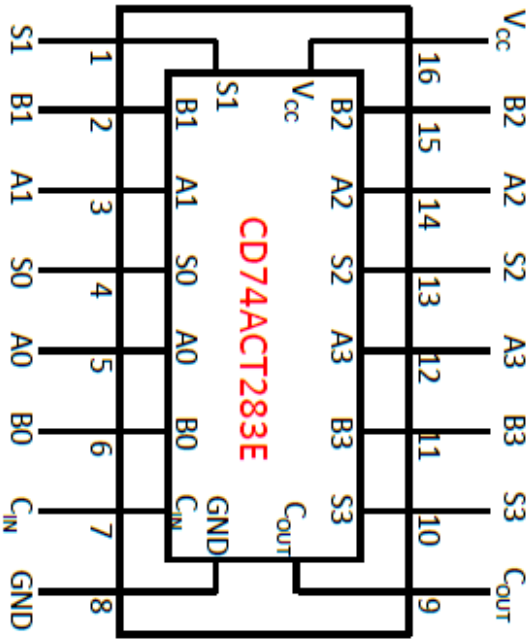


Figure 7. LED terminals

Table 7. Output voltage measurements.

CD74ACT283E Output Pin	Voltage with LED connected	Voltage with LED disconnected
S0 (logic 1 state)		
S0 (logic 0 state)		
C_{OUT} (logic 1 state)		
C_{OUT} (logic 0 state)		





Digital Writer

- DIO 0
- DIO 1
- DIO 2
- DIO 3
- DIO 4
- DIO 5
- DIO 6
- DIO 7

Digital Reader

- DIO 8
- DIO 9
- DIO 10
- DIO 11
- DIO 12
- DIO 13
- DIO 14
- DIO 15

- LED 0
- LED 1
- LED 2
- LED 3
- LED 4
- LED 5
- LED 6
- LED 7

- +5V
- GROUND