Digital Design Fundamentals
CSC 120
Instructor: Bruce Carlton
Computer Science Department

Mesa Community College
1833 W. Southern Avenue
Mesa, Arizona 85202

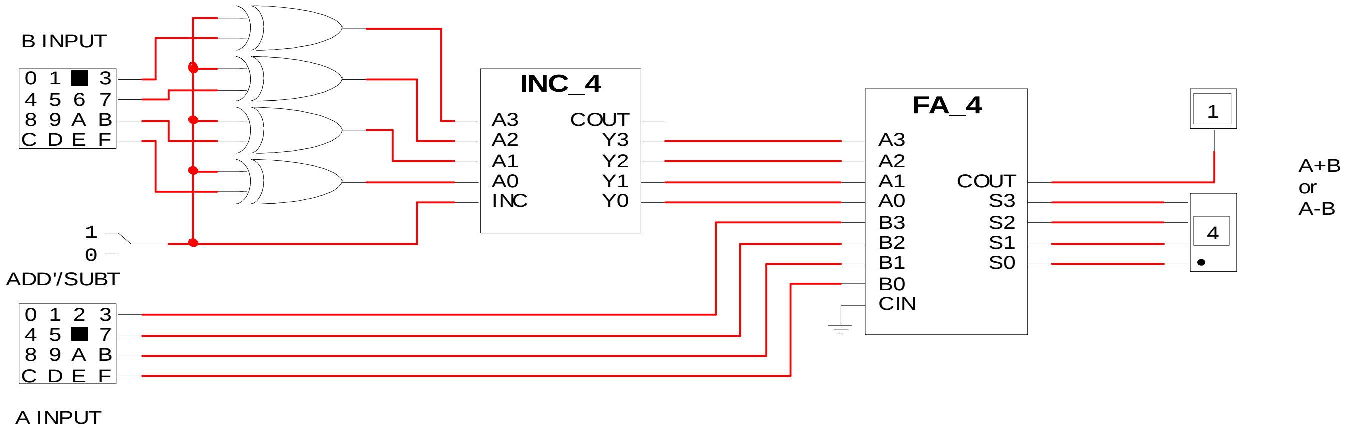
Software Lab #3
This Assignment Worth: 40 Points
Submit by either: Electronic Submission (Email: BruceatMCC@msn.com) or Hardcopy

Required Files (6): classlib.circ
Containing:
Add_Subt
Shifter_Sign_LW5
Multiply_Sign_LW5
Divide_Sign_LW5

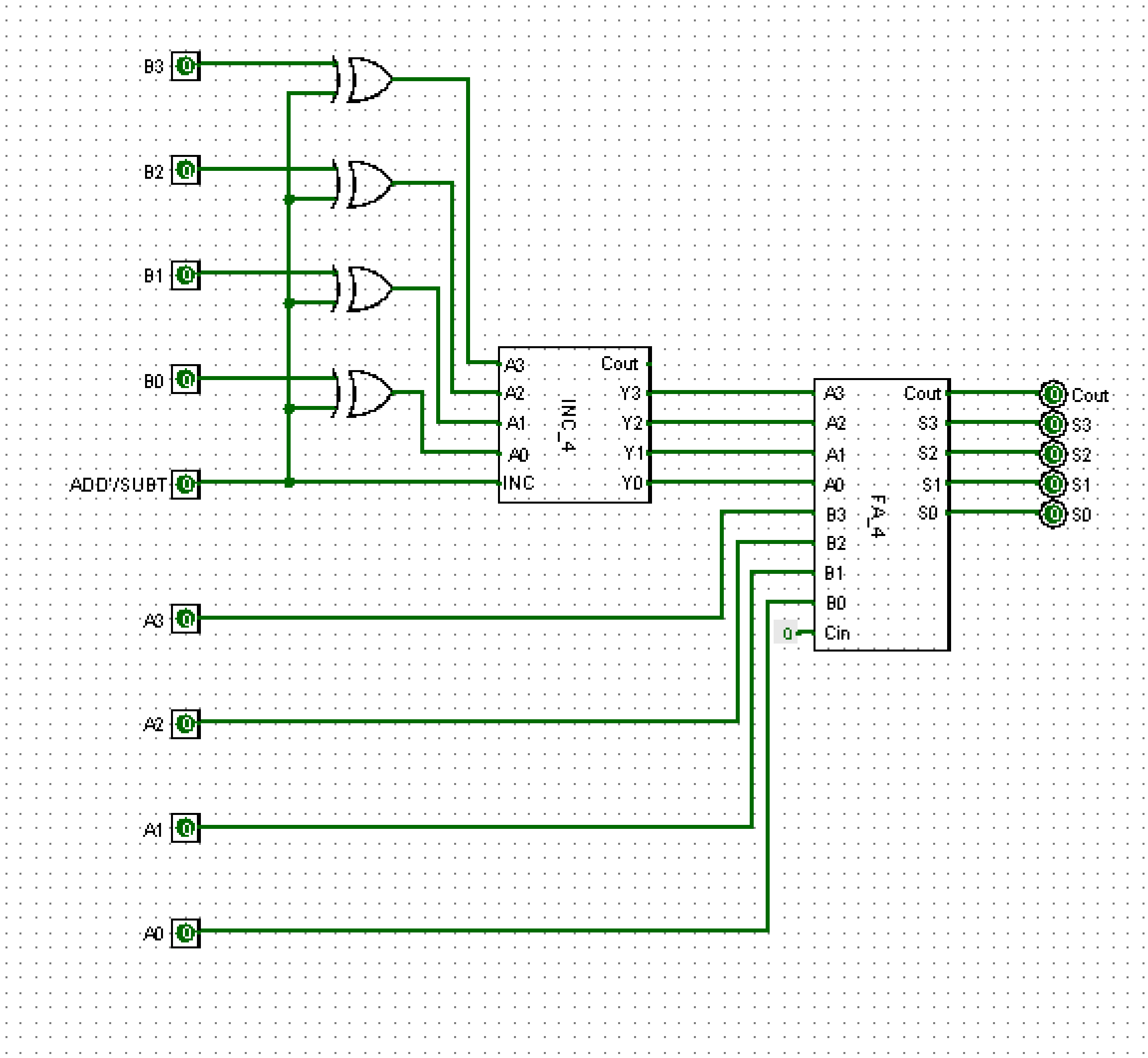
Lab Report: Lab3.doc

Task #1 - (10 Points) Building a Signed Add_Subtract Circuit

Modify the two's complement circuit (which used the 4-bit increment) built in SW Lab #2 to perform an Add/Subtract (taking the two's complement) function. Remember that taking the two's complement of a binary number in two's complement representation is essentially finding the negative value of that number. Then, an adder can be used to perform the necessary subtraction operation. When the ADD'/SUBT input is low, the B input is passed through the INC_4 and added to the A input for the A+B operation. When the ADD'/SUBT input is high, the B input is 2's complemented and added to perform the A–B operation. Here is the circuit in LogicWorks format, and Logisim drawn versions of the circuits in this lab can be found in the associated image files:



Mesa Community College
1833 W. Southern Avenue
Mesa, Arizona 85202

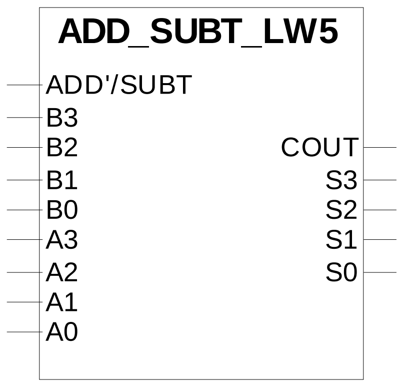


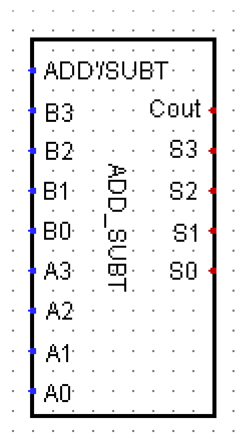
Build this circuit in your classlib file. Test this circuit. Name it as ADD_SUBT. In your lab report include:

- A narrative description of how this circuit works
- Input hexadecimal values A input = C, B input = 2 with each of the Add and Subtract operations

Now, test the device symbol. In your lab report include:

- Input hexadecimal values A input = D, B input = 1 with each of the Add and Subtract operations

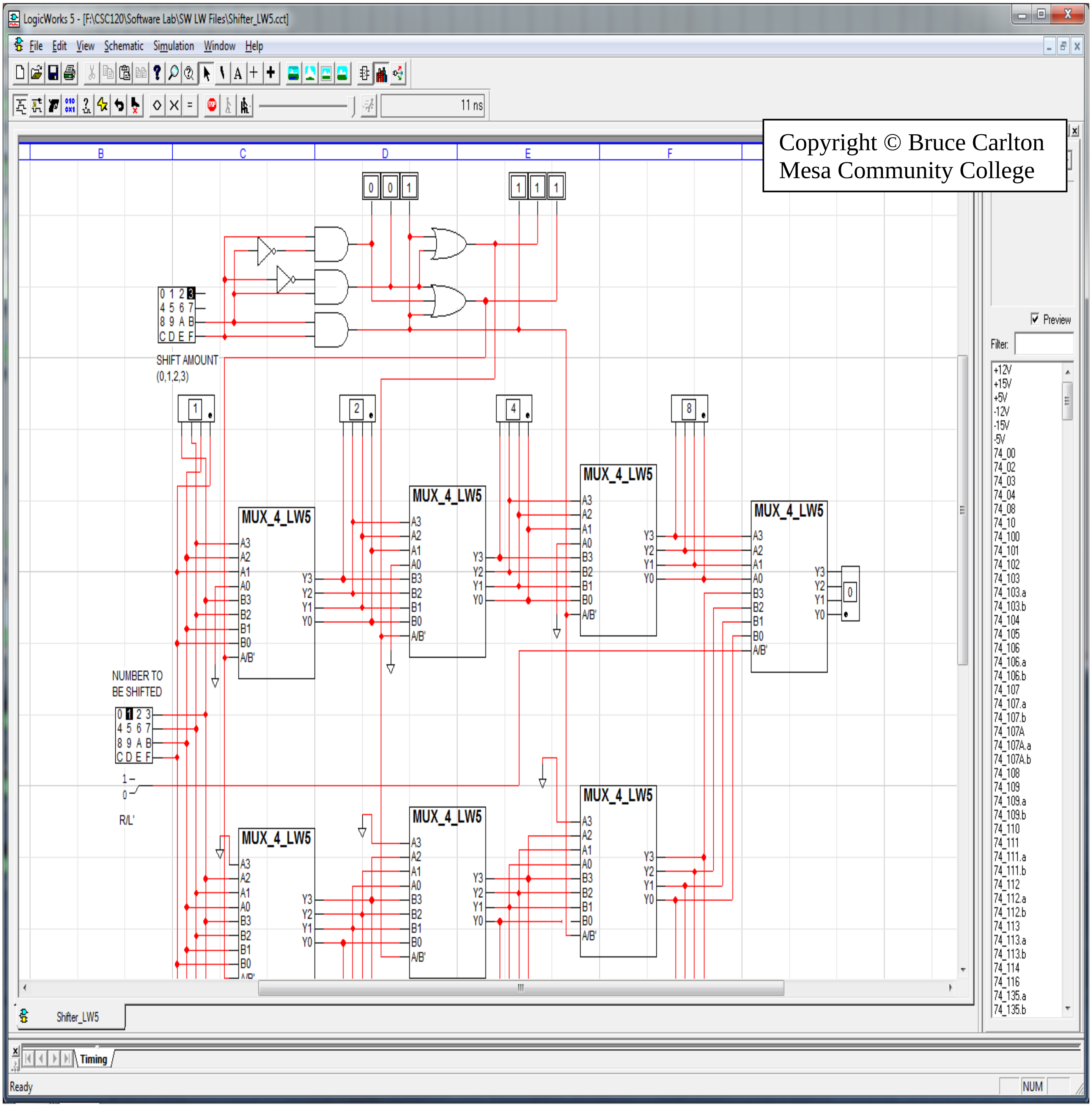




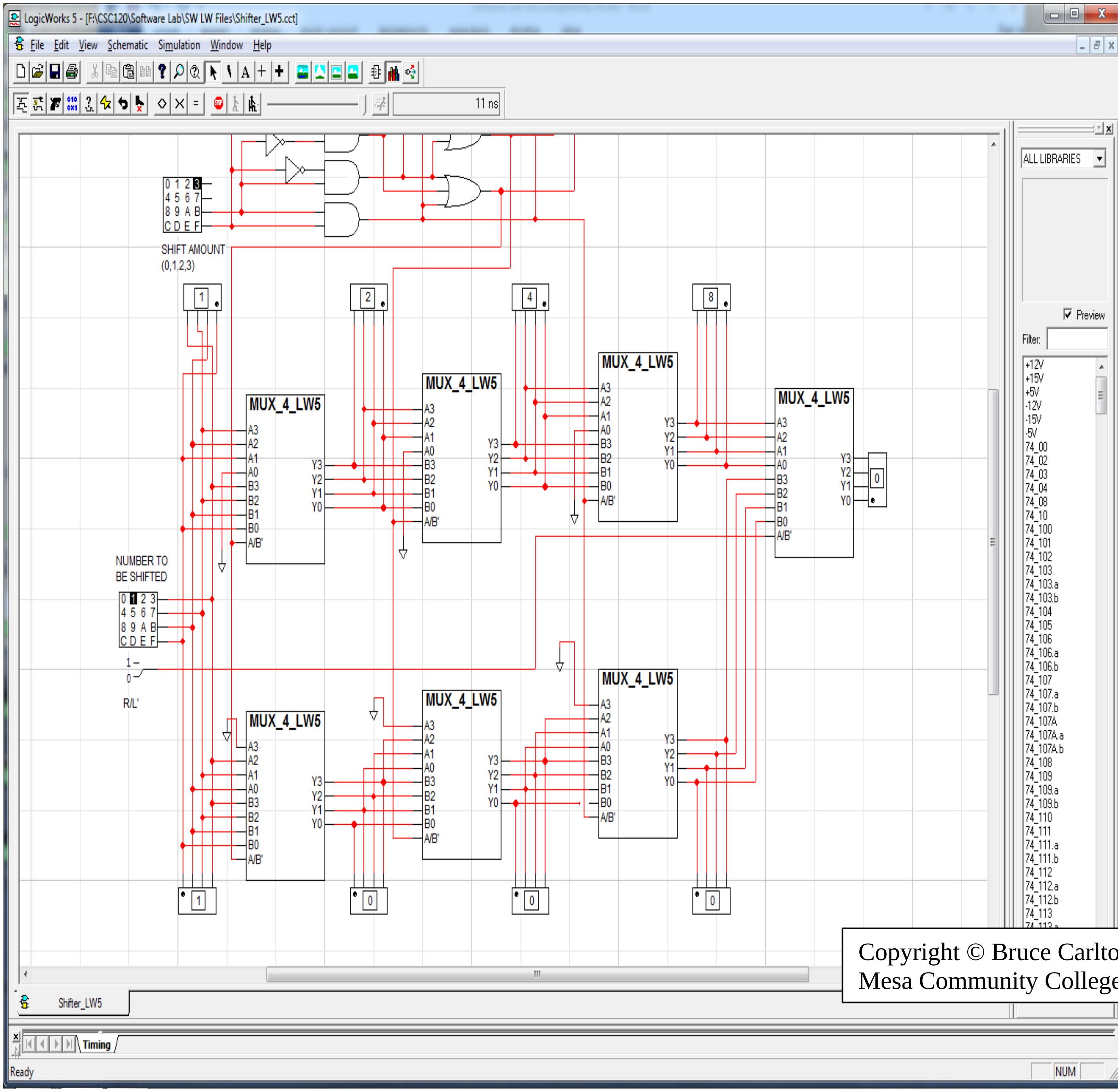
This ADD/SUBT circuit is the start of our Arithmetic Logic Unit (ALU). ALU's perform binary arithmetic (add, subtract, multiplication, and division) functions. Include all detailed design to support your circuit (schematics, truth tables, k-maps, equations, etc).

Task #2 - (10 Points) Building an Unsigned Shifter Circuit

The shifter circuit you will design will shift a 4-bit signed binary number either to the right or to the left. Remember shifting to the right moves all bits to the right, filling in zeroes on the left and allowing bits that fall off the right side to be lost. Shifting to the left moves all bits to the left, filling in zeroes on the right and allowing bits that fall off the left to disappear. These are division and multiplication as long as no ones fall off the end. This circuit is built using the MUX_4 device symbols that you previously built in SW Lab #2. Also, note that any shift (either right or left) of more than 3-bits on a 4-bit number will result in 0000. Remember that we backfill the vacant flip-flop input with 0's (shift right, LSB to MSB) and 0/1 (shift left, MSB to LSB depending upon the sign bit) in a signed shifter, but we are building an unsigned shifter here, so it always fills in zeroes. This circuit has two inputs: the 4-bit number to be shifted and the number of shifts (SHAMT). When the input L/R' is High, the input will be shifted to the left (LSB → MSB) and when the L/R' is Low, the input will be shifted to the right (MSB → LSB). Here's the circuit (the control line is reverse named in the LogicWorks diagram):



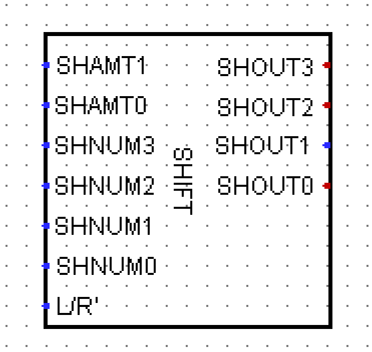
Schematic continued next page.



Build this circuit as a New Circuit in your Classlib file, naming it SHIFTER. Test this circuit. In your lab report include:

- A narrative description of how this circuit works

- Input values L/R' = 1, Number to be Shifted = 1, Shift Amount = 3. Also, L/R' = 0, Number to be Shifted = 8, Shift Amount = 3.



Now, test the device symbol. In your lab report include:

- A narrative description of how this circuit works
- Input values L/R' = 1, Number to be Shifted = 1, Shift Amount = 2. Also, L/R' = 0, Number to be Shifted = 8, Shift Amount = 2.

Include all detailed design to support your circuit (schematics, truth tables, k-maps, equations, etc).

<u>Task #3 - (10 Points) Building a Signed Multiplier Circuit</u>

On to the 4 x 4 signed multiplier circuit! This will multiply two signed 4-bit binary numbers and get an 8-bit signed output. For this circuit, you will use the 4-bit adder FA_4, and the 4-bit multiplexer MUX_4. The algorithm for the 4 x 4 multiplier is to first add, and then shift that will accumulate partial products. Note that here are no clocks! You will use two FA_4 adders to make an 8-bit adder. The algorithm is to move from the LSB to the MSB of the multiplier, note that if the bit is 1 the multiplicand is entered, if the bit is 0 all 0's are entered. Partial sums are formed from the 2nd partial multiplication through the 4th multiplication. Also, for the multiplication of 2 4-bit numbers, the solution may be as many as 8-bits.

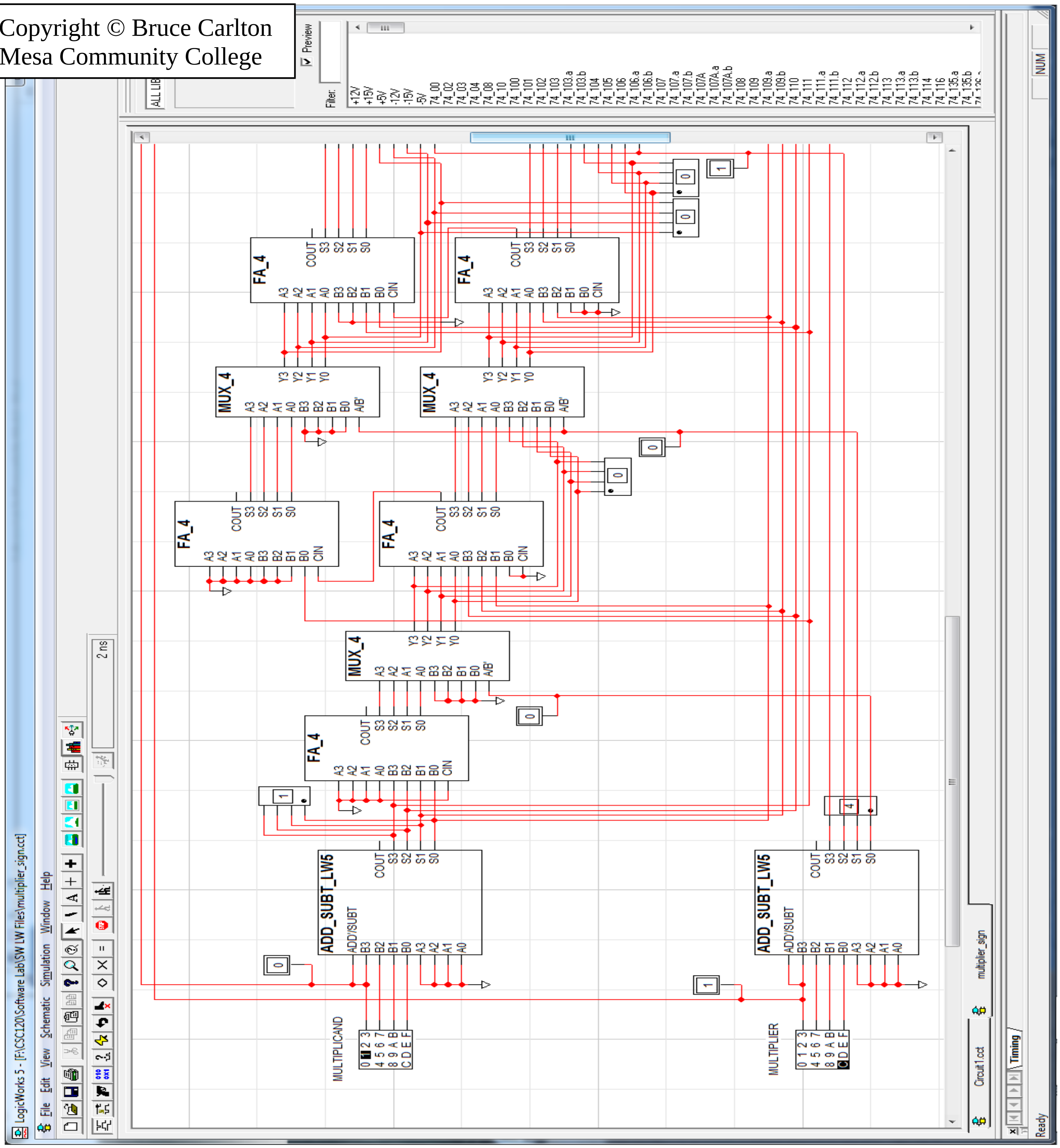
Example:

```
      0100
    x 0110
   -----------
      0000
     0100   → partial sum = 01000
    0100    → partial sum = 011000
   0000     → partial sum = 0011000
 ----------------
(0)0011000
```

Instructor: Bruce Carlton
Computer Science Department

Mesa Community College
1833 W. Southern Avenue
Mesa, Arizona 85202

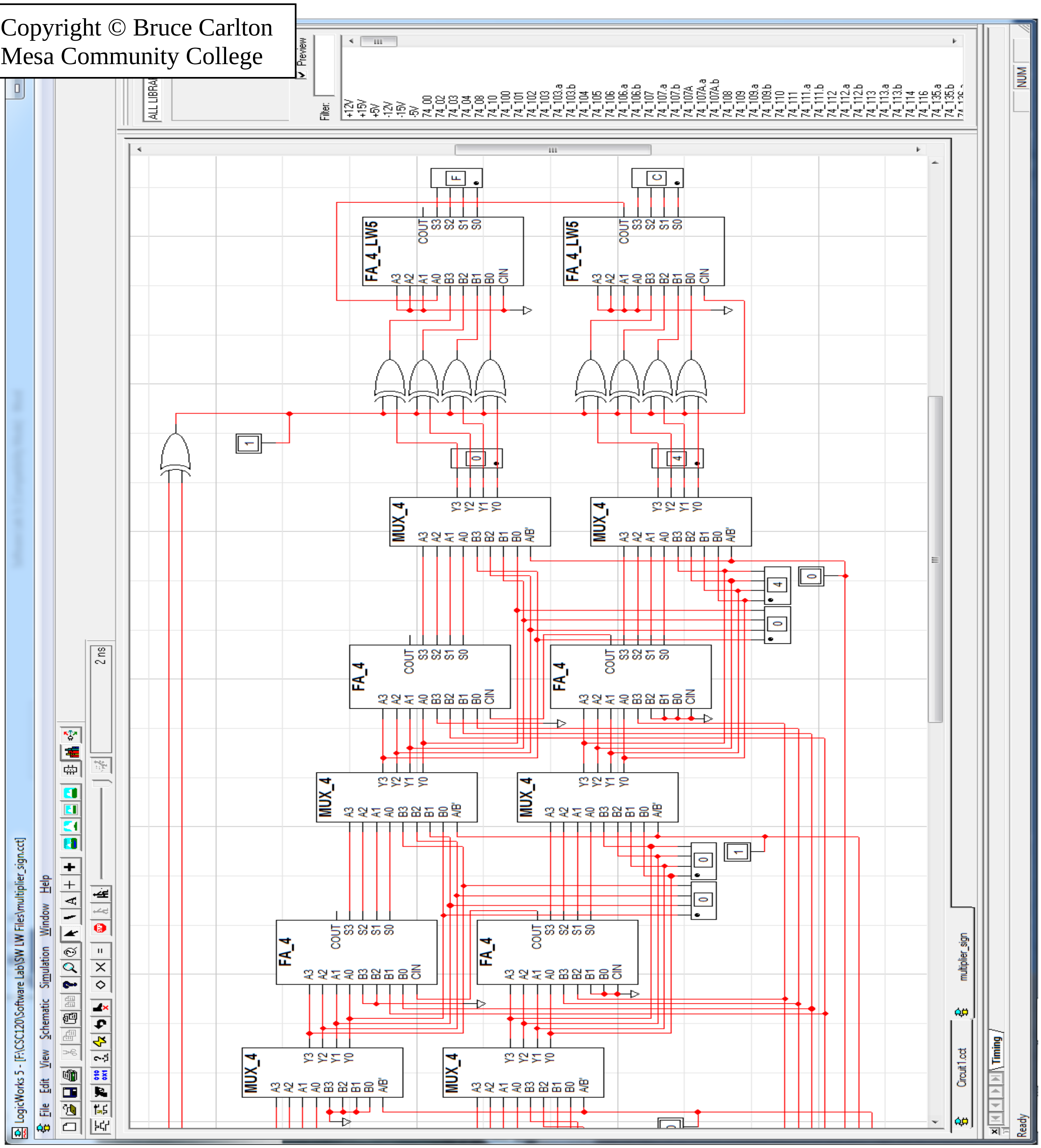
Here is the circuit:



Schematic continued next page.

Instructor: Bruce Carlton
Computer Science Department

Mesa Community College
1833 W. Southern Avenue
Mesa, Arizona 85202



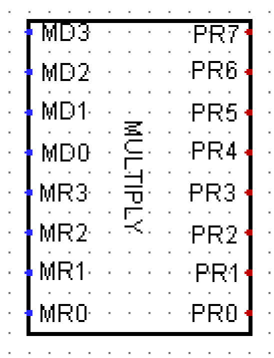
Build this circuit in your classlib file. Test this circuit. Name it as MULTIPLY. In your lab report include:

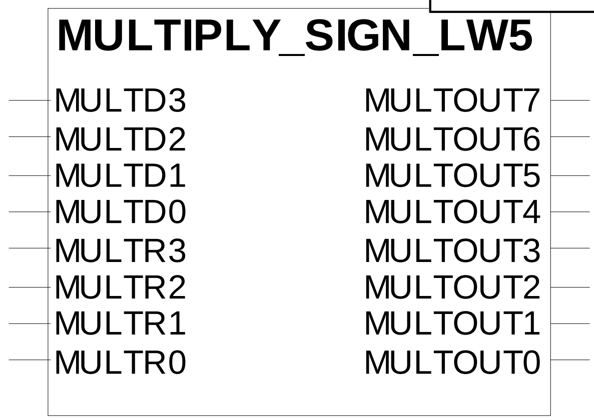
- A narrative description of how this circuit works
- Input values Multiplicand = 5, Multiplier = 6

Now, test the device symbol. In your lab report include:

- A narrative description of how this circuit works
- Input values Multiplicand = 4, Multiplier = 9

Include all detailed design to support your circuit (schematics, truth tables, k-maps, equations, etc).

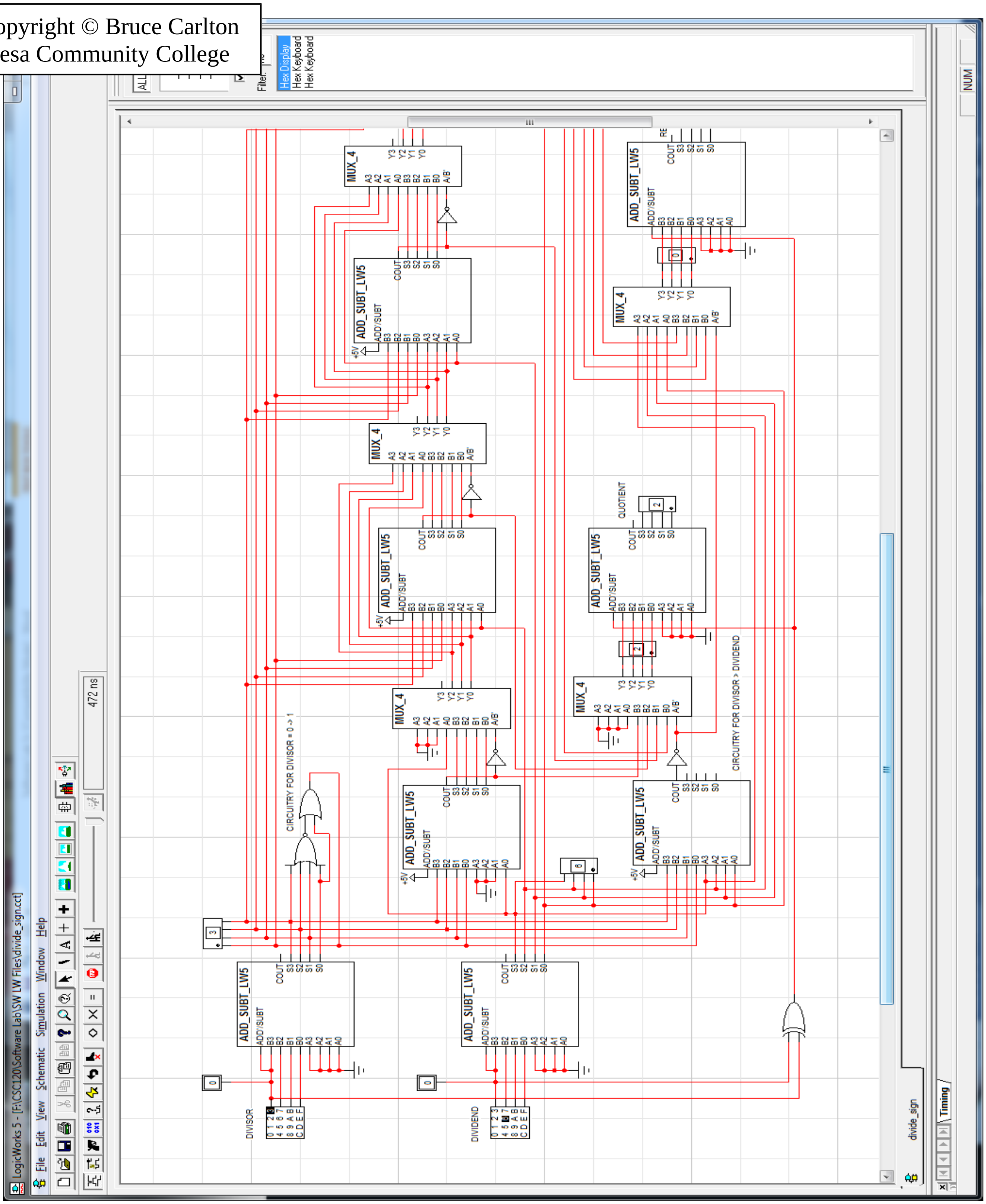




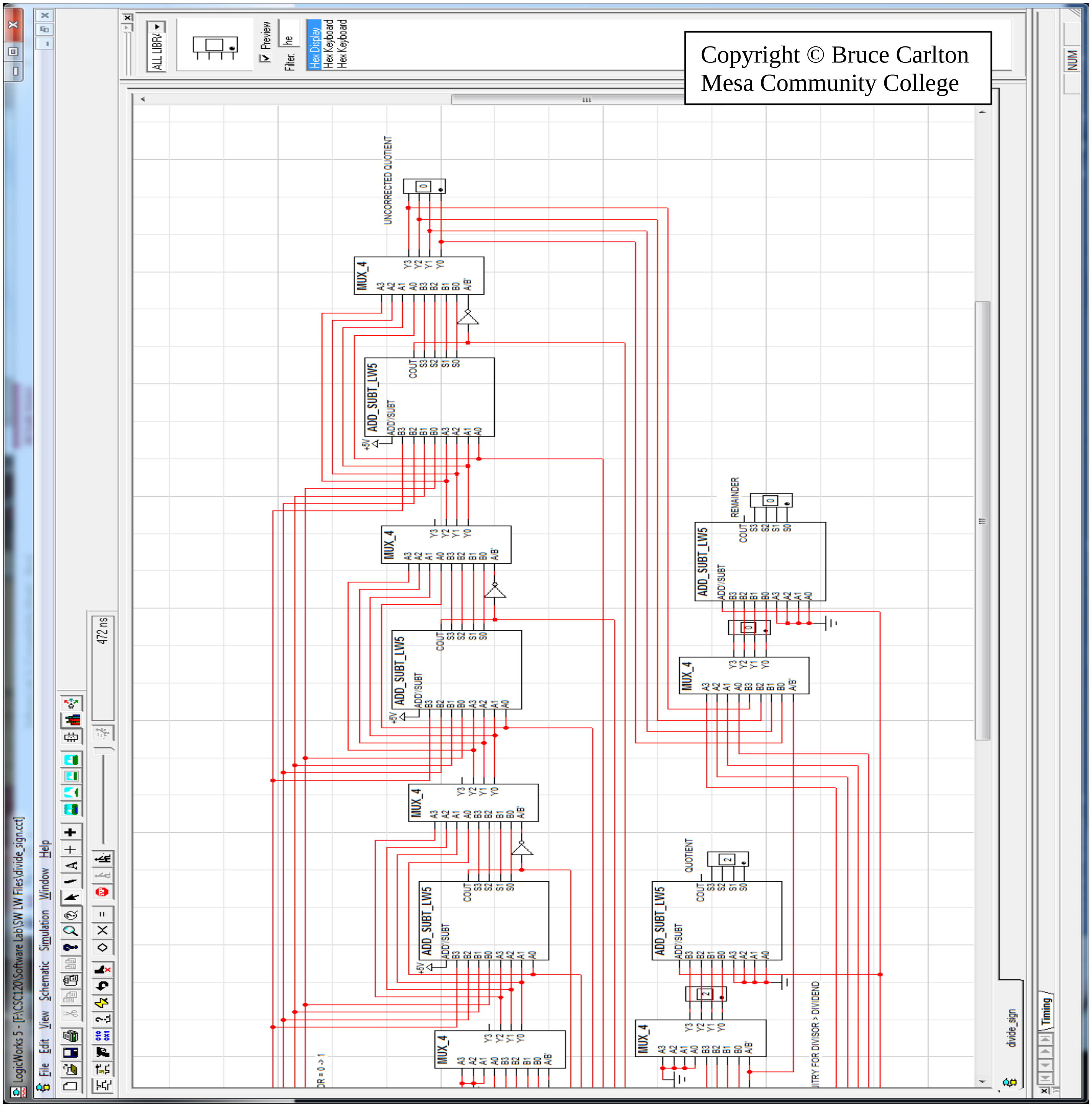
Task #4 - (10 Points) Building the Signed Divide Circuit

Now on to the signed divider circuit! This will divide two signed 4-bit binary numbers and get a 4-bit signed quotient output and a 4-bit signed remainder output. For this circuit, you will use the ADD_SUBT_LW5 and MUX_4 device symbols. The algorithm for the 4 x 4 divider is to first subtract, and then shift that will accumulate partial products. Note that here are no clocks! There also is provision to ensure that there cannot be a division by zero and also circuitry for when the divisor is greater than the dividend. Also notice that it incorrectly divides a negative number and a positive number, giving a negative remainder. All remainders are positive in standard calculations.

Mesa Community College
1833 W. Southern Avenue
Mesa, Arizona 85202



Schematic continued next page.


Copyright © Bruce Carlton
Mesa Community College
UNCORRECTED QUOTIENT
REMAINDER
QUOTIENT
MUX_4
ADD_SUBT_LW5

Build this circuit in your classlib file. Test this circuit. Name it as DIVIDE. In your lab report include:

- A narrative description of how this circuit works
- Input values Divisor = 5, Dividend = F

Now, test the device symbol. In your lab report include:

- A narrative description of how this circuit works
- Input values Divisor = 4, Dividend = D
- Include all detailed design to support your circuit (schematics, truth tables, k-maps, equations, etc).