SYSC2310 – LAB 4 REPORT

1.0 EXERCISE 1: DESIGN A SIMPLE 4-BIT ALU

1.1 Description of Circuit's Operation

The circuit in Exercise 1 is a simple 4-bit arithmetic logic unit (ALU). The circuit's inputs are two 4-bit inputs, A and B, three 1-bit inputs representing the operation codes, $S_0, S_1, and\ S_2$, and a carry-in 1-bit input. The circuit's outputs include a 4-bit binary output representing the result, a 1-bit carry-out, a sign bit, and an overflow bit. This is shown in Figure 1.

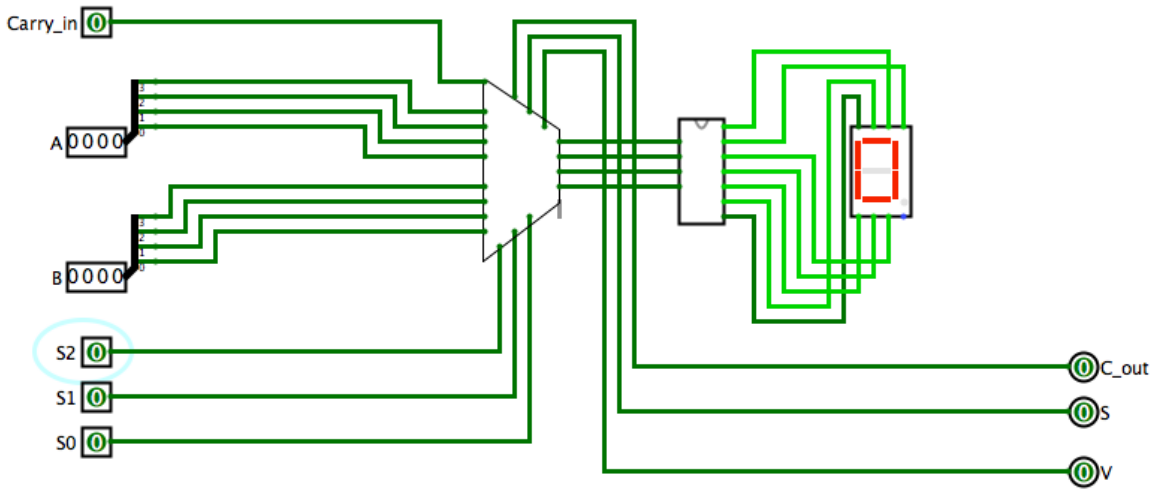


Figure 1. Exercise 1 main circuit diagram.

The circuit can perform 4 logic functions and 4 arithmetic functions. The logic functions include AND, OR, XOR, and XNOR operations. The arithmetic functions include A+1, A+B, A-B, and B-A. Seven multiplexers were used to switch between the outputs of the 8 functions accordingly with the values of $S_0, S_1, and\ S_2$, shown in Figure 2. The circuit includes many subcircuits to perform these functions.

The logic functions were very straightforward to design, as they all involved passing all of the pairs of bits into a corresponding logic gate, depending on the logic function in question and all follow the same pattern. For example, shown in Figure 2 is the AND logic function. This function took in a pair of 4-bit inputs and then passed the nth bit from both sets into an AND gate and outputted it as the nth output, where n is the index for a 4-bit number. As stated before, all of the logic functions follow the same pattern and the circuit diagrams for the OR, XOR, and XNOR functions are found in Figures 3, 4, and 5, respectively.

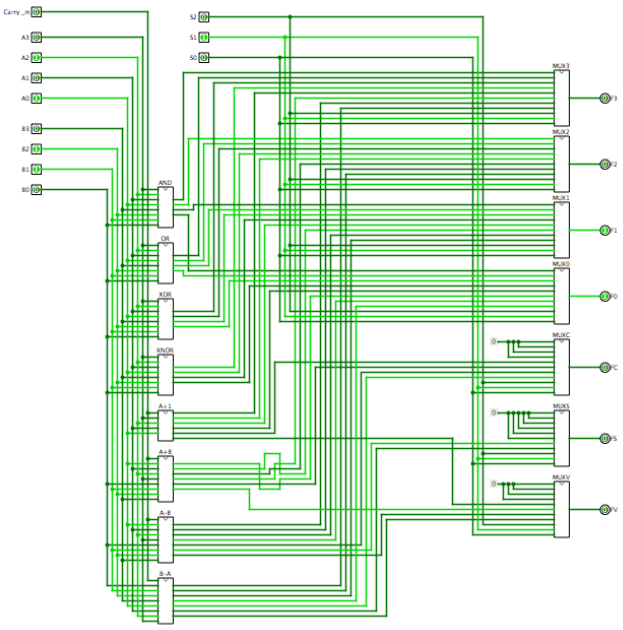


Figure 2. Multiplexers used to switch between the outputs of the 8 functions.

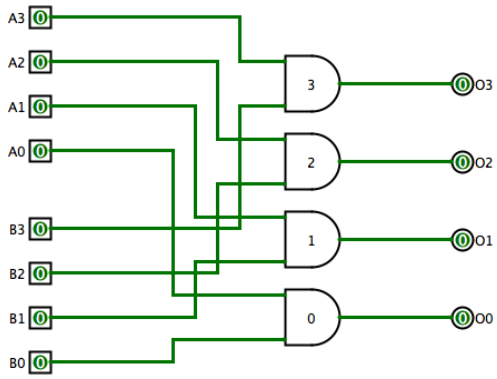


Figure 2. Circuit diagram of AND logic function

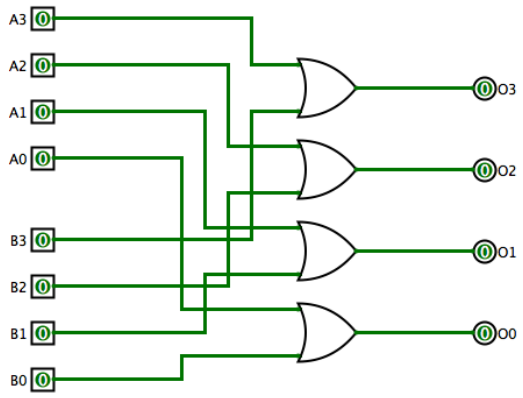


Figure 3. Circuit diagram for OR function

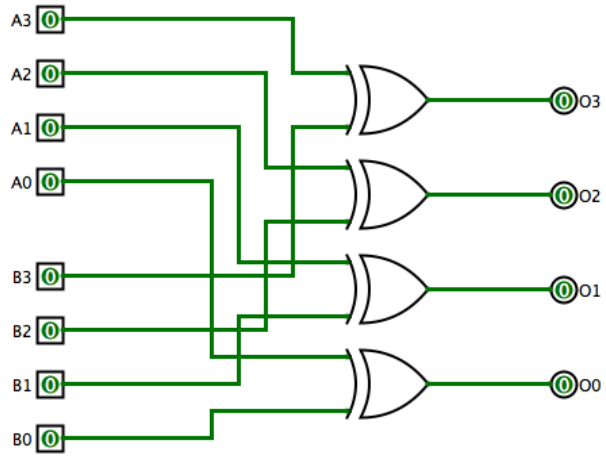


Figure 4. Circuit diagram for XOR function

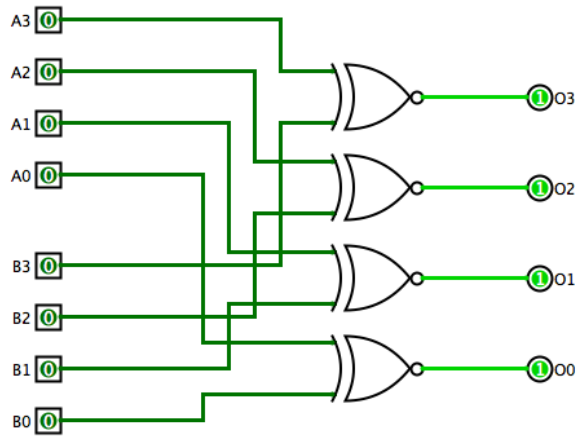


Figure 5. Circuit diagram for XNOR function

The arithmetic functions, while not identical in design to each other like the logic functions, built off of each other. The core of the functions consisted of the 4-bit full adder, made in Lab 1, with overflow detection added, shown in Figure 5.

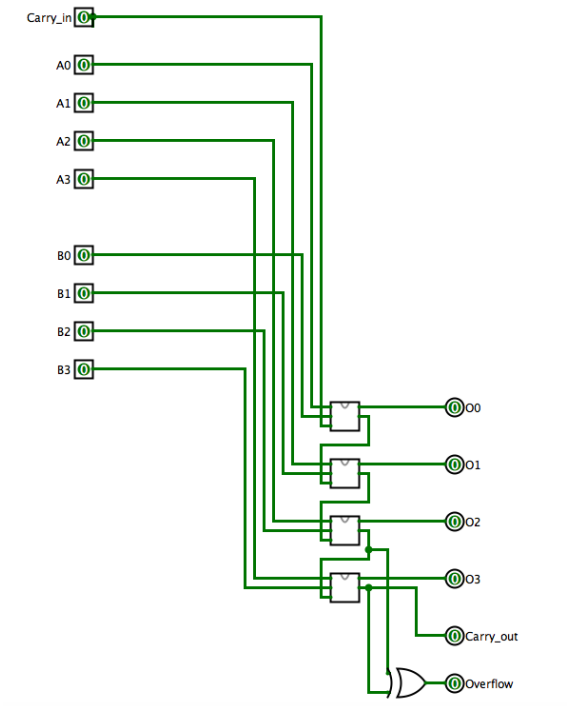


Figure 5. Full adder with overflow detection circuit

The A+1 function uses the full adder, taking in 4 bits and a carry bit as inputs along with constants representing the number 1 in binary, shown in Figure 6.

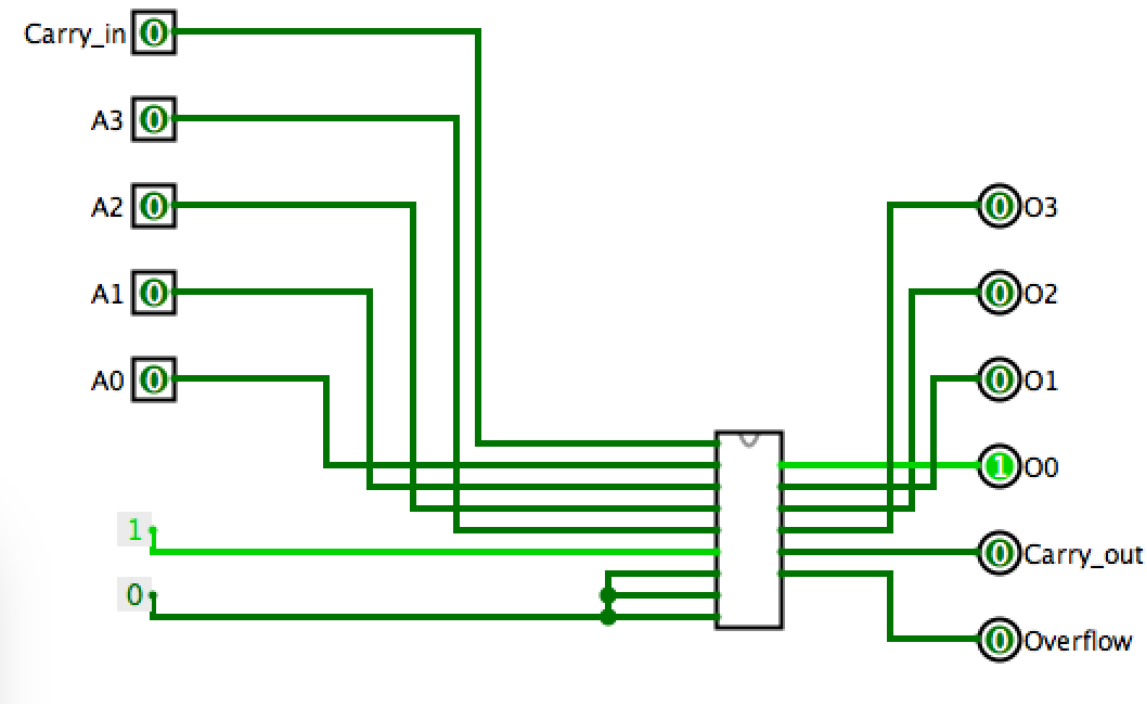


Figure 6. Function A+B circuit.

Function A+B consists only of the full adder, as shown in Figure 5.

Function A-B takes in 2 4-bit inputs, a 1-bit carry, and outputs a 4-bit number, a 1-bit carry-out, an overflow bit, and a sign bit, shown in Figure 7. The function utilizes 3 subcircuits: a two's complement function, which flips the bits and adds 1, a 4-bit full adder, and a "negative number corrector", which turns a 2s complement number back to binary, if need be; the diagrams for the 2s complement function and the negative number corrector are shown in Figures 8, and 9, respectively. The function converts number B from binary to its 2s complement and then adds it with A using the full adder. Then, if the result is in 2s complement format, it is turned back into binary, however the function still takes note that the number is negative, using the sign bit.

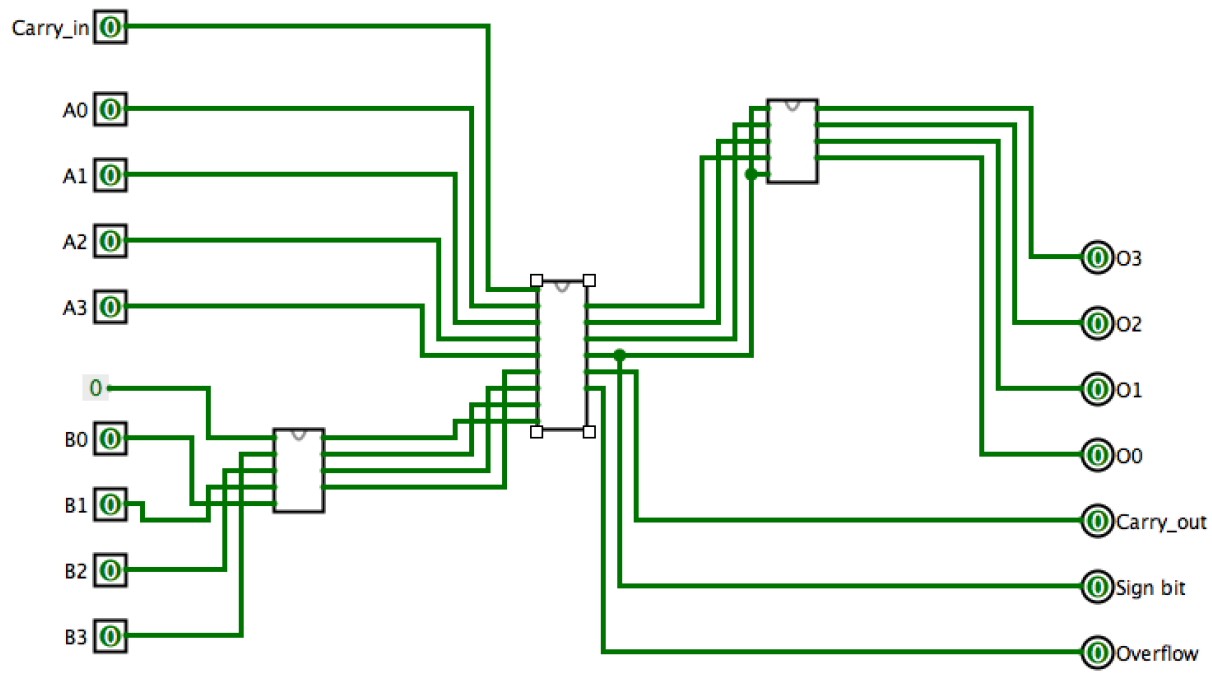


Figure 7. Function A-B circuit diagram

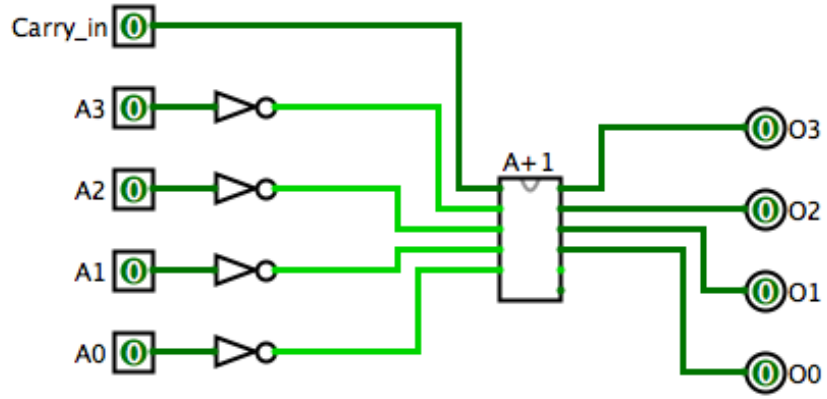


Figure 8. Two's complement circuit diagram

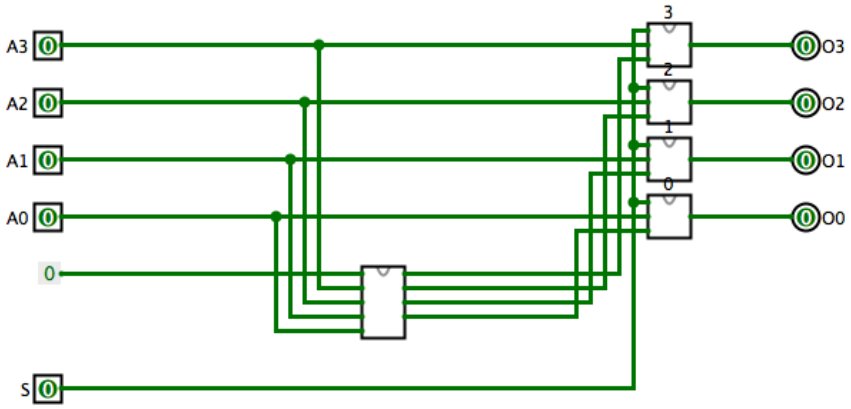
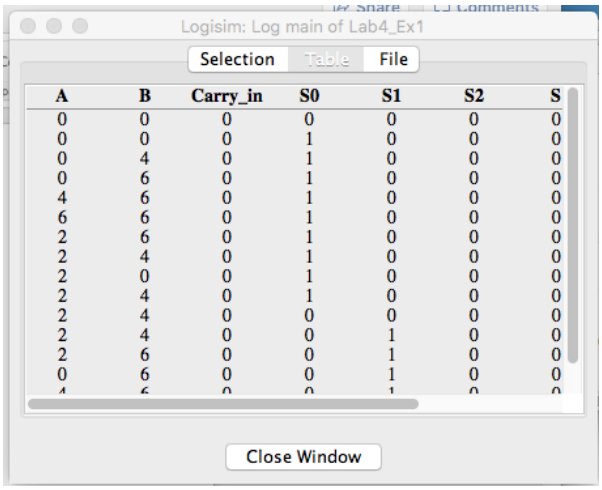


Figure 9. Negative number corrector circuit diagram

Function B-A uses the same function as A-B.

1.2 Verification of Circuit's Operation

The circuit's operation has been verified through the use of Logisim's logging function, shown in Figure 10, by comparing actual outputs with their expected outputs.



A	B	Carry_in	S0	S1	S2	S
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	4	0	1	0	0	0
0	6	0	1	0	0	0
4	6	0	1	0	0	0
6	6	0	1	0	0	0
2	6	0	1	0	0	0
2	4	0	1	0	0	0
2	0	0	1	0	0	0
2	4	0	1	0	0	0
2	4	0	0	0	0	0
2	4	0	0	1	0	0
2	6	0	0	1	0	0
0	6	0	0	1	0	0

Figure 10. Logisim logging file used for verification

1.3 Analysis of the Circuit's Gates

The following section will complete the following tasks assigned in Exercise 1:

(i) Compute the total number of gates, assuming that all the gates have the same implementation area

(ii) Compute the maximum delay of the overall ALU, assuming that all the gates have the same delay

Each of the 4 logic functions have 4 gates, which is a **total of 16 gates and a max delay of 1 gate.**

Both A+1 and A+B use the full adder only. Since the full adder uses four 1-bit adders, which each use 5 gates and have a max delay of 2 gates, the full adder uses 20 gates and has a max delay of 8 gates and adding the overflow detection, this sums to 21 gates and a max delay of 9 gates. **Thus, A+1 and A+B have a total of 42 gates and each have a max delay of 9 gates.**

Both A-B and B-A are identical. They use the full adder, the two's complement function, and the negative number corrector function. The two's complement function uses function A+1, and thus has 21 gates and a max delay of 9 gates. The negative number corrector function uses the two's complement function and four 2-to-1 1-bit MUX, which each have 3 gates and a max delay of 2 gates. Thus, the 4 MUXes have a total of 12 gates and a max delay of 2 gates. Therefore, combined with the two's complement function, the negative number corrector has a total of 33 gates and a max delay of 11 gates. Therefore, combining the full adder, the two's complement function, and the negative number corrector, **functions A-B and B-A have a total of 150 gates and each have a max delay of 29 gates.**

There are 7 8-to-1 1-bit MUX that each have 9 gates and a max delay of 2 gates. Therefore **the 7 MUX have a total of 63 gates and each have a max delay of 2 gates.**

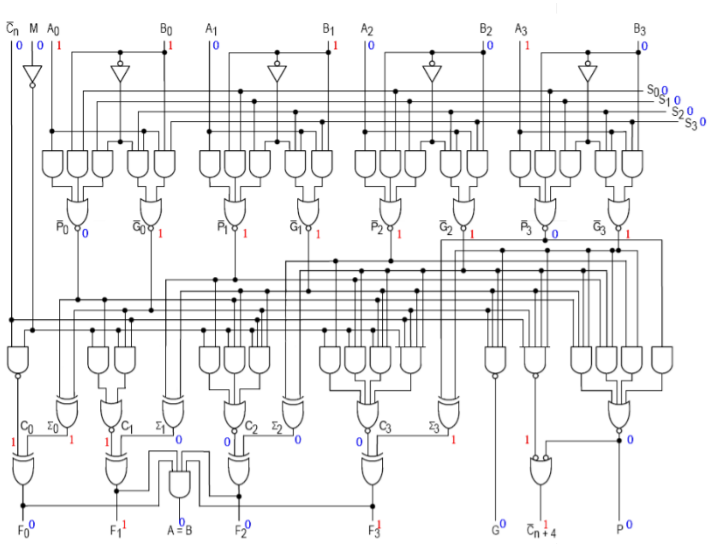
The 7-segment decoder has a total of 22 gates and a max delay of 2 gates.

Summing up all of this data, recognizing that the arithmetic and logic functions are parallel to each other and thus their max delays to not add, so the largest one is taken. Therefore, the total number of gates is 293 and the maximum delay is 34 gates.

2.0 EXERCISE 2: ANALYZING THE 43181 ALU

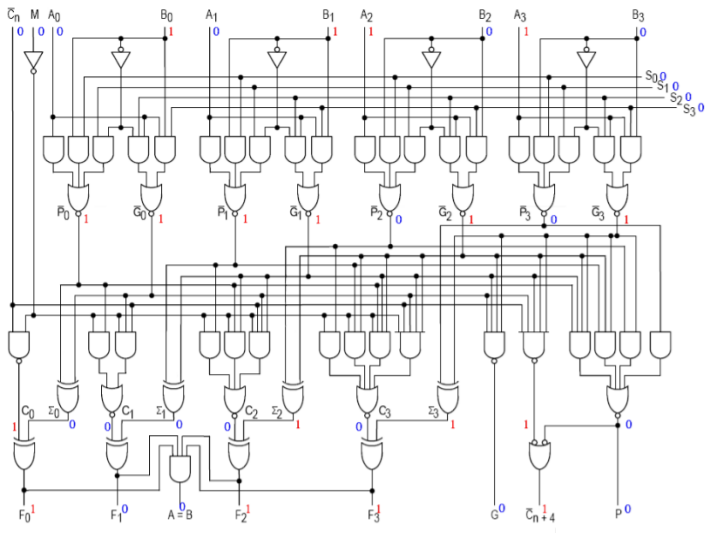
1. [S3 S2 S1 S0] = [L L L L], M=L, thus Cn = L, therefore, performing F=A plus 1, for A=[1001] and A=[1100]. Verify the correctness of the output

   When A = [1001]:



F = [1 0 1 0] = A + 1 -> CORRECT

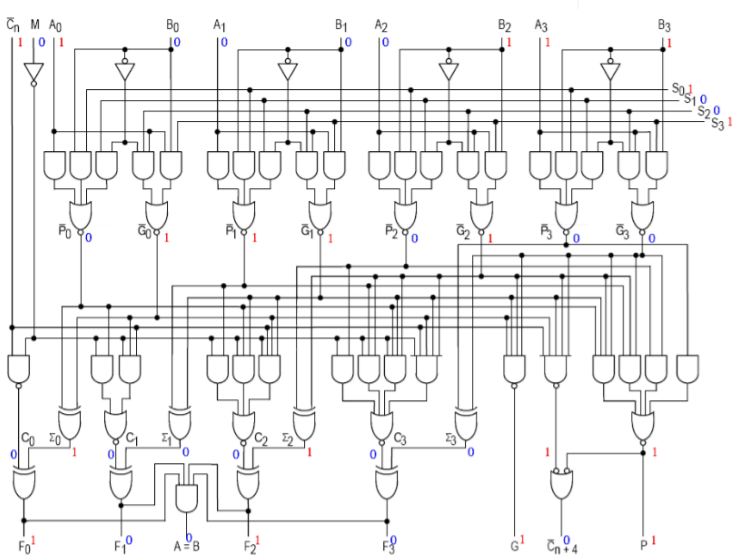
When A = [1100]:



F = [1 1 0 1] = A + 1 -> CORRECT

Therefore, the output is correct

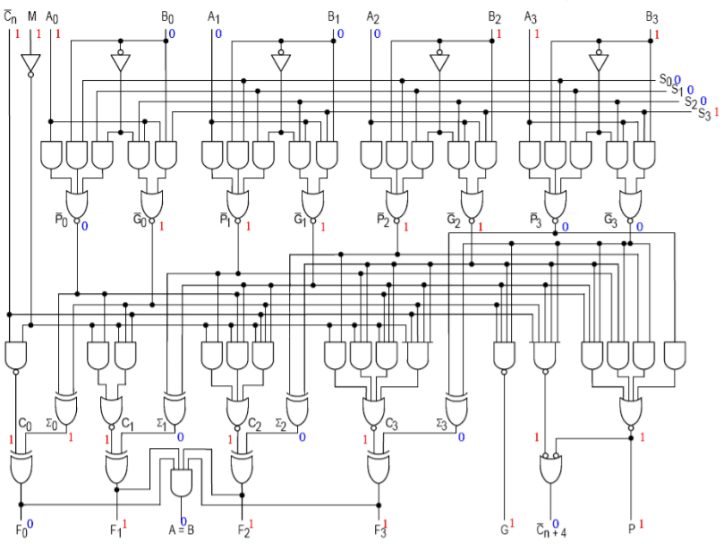
2. [S3 S2 S1 S0] = [H L L H], M=L, thus arithmetic operation and Cn = H, therefore, performing F = A plus B, for A=[1001] and B=[1100]. Verify the correctness of the output.



F = [0 1 0 1], which is NOT equal to A plus B which is equal to [1 0 1 0 1]
However, this is because this result is unable to be stored in the 4-bits representing F and so the carry bit is high.

Nonetheless, strictly speaking in the case of the question, and ignoring the carry output, the (expected) output is NOT CORRECT, because it should have been expected to be equal to [0101].

3. [S3 S2 S1 S0] = [H L L L], M = H, thus logical operation and Cn = H, therefore, performing F = A' + B, for A = [1 0 0 1] and B = [1 1 0 0]. Verify the correctness of the output.



F = [1 1 1 0] = A' + B

Therefore, the output is CORRECT.

4. Find the total number of gates in the design of the 74181 ALU, assuming that al the gates have an equivalent area.

   - 4 NOT gates
   - 20 + 14 + 1 = 35 AND gates
   - 8 + 4 = 12 NOR gates
   - 4 + 4 = 8 XOR gates
   - 1 OR gate
   - 3 NOR gates

   Therefore, there is a total of 59 gates.

5. Find the maximum delay of the 74181 ALU, assuming that all the gates have an equivalent delay.

   The maximum delay of the ALU is 7 gates.