Michael Campos
Prof. Li

05/11/2020
CSE222

Final Project Part I

1. To convert a 4-bit signed/magnitude binary number to two's complement you would have 4 inputs: A, B, C, D to represent the signed binary number. A is the msb and D is the lsb. For the output, you would have 4 outputs: Y0, Y1, Y2, Y3 to represent two's complement. Y0 is the msb and Y3 is the lsb. First take A and connect it to an AND gate with a constant high voltage. We do this to check if the signed binary number is negative. If A*1 is 0 then we return A = Y0, B = Y1, C = Y2, and D = Y3 because a positive number in signed/magnitude is the same in two's complement format. If A*1 is 1,then the signed binary number is negative, so we negate B, C, and D. Then we can use XOR and AND gates to build half adders and full adders to add 1 to the lsb and use it's cout to add C, then repeat for B and A. When we add A and B's cout, we discard A's cout since were only working in a 4 bit system. Finally, you have your signed binary number in two's complement.

2.

A	B	C	D	Y_0	Y_1	Y_2	Y_3
0	0	0	0	0	0	0	0
0	0	0	1	0	0	0	1
0	0	1	0	0	0	1	0
0	0	1	1	0	0	1	1
0	1	0	0	0	1	0	0
0	1	0	1	0	1	0	1
0	1	1	0	0	1	1	0
0	1	1	1	0	1	1	1
1	0	0	0	0	0	0	0
1	0	0	1	1	1	1	1
1	0	1	0	1	1	1	0
1	0	1	1	1	1	0	1
1	1	0	0	1	1	0	0
1	1	0	1	1	0	1	1
1	1	1	0	1	0	1	0
1	1	1	1	1	0	0	1

3.

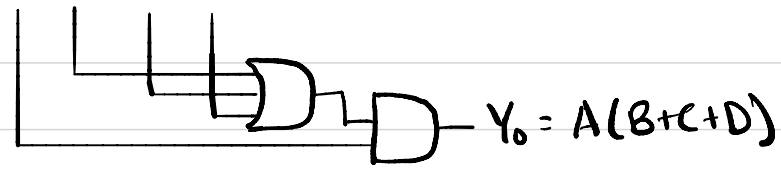
$$Y_0 = A(B+C+D)$$

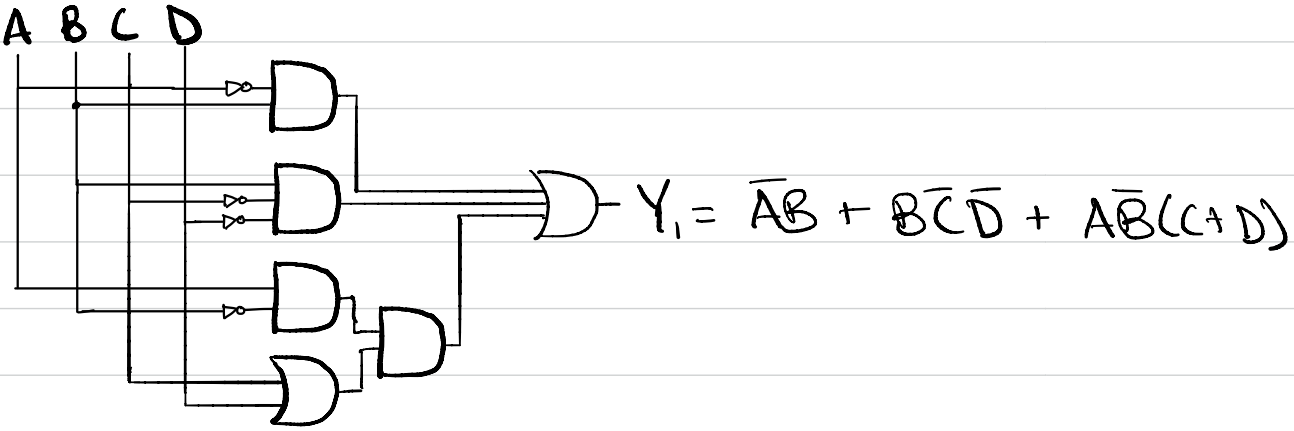
$$Y_1 = \bar{A}B + B\bar{C}\bar{D} + A\bar{B}(C+D)$$

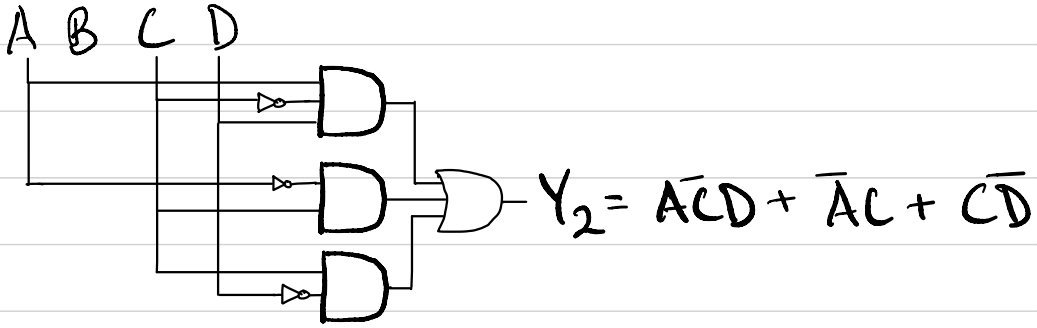
$$Y_2 = A\bar{C}D + \bar{A}C + C\bar{D}$$

$$Y_3 = D$$

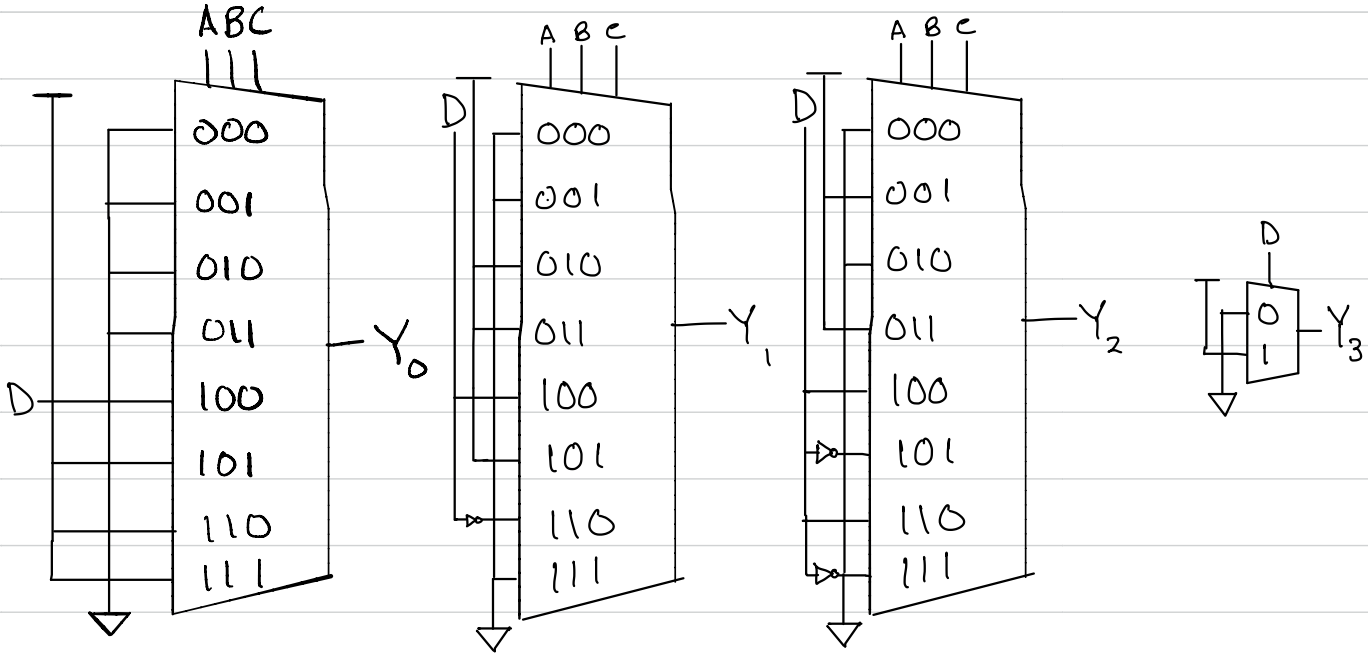
4. A B C D


$Y_0 = A(B+C+D)$


A B C D
$Y_1 = \bar{A}B + B\bar{C}\bar{D} + A\bar{B}(C+D)$


A B C D
$Y_2 = A\bar{C}D + \bar{A}C + C\bar{D}$

$D - Y_3 = D$


ABC
000
001
010
011
100
101
110
111
D
Y_0
A B C
D
Y_1
Y_2
D
0
1
Y_3

Michael Campos
Prof. Li

May 13, 2020
CSE222

Final Project Part II

1. The way I implemented my MIPS program to convert a 32-bit signed binary number to its two's complement is I first prompt the user to enter the 32-bit signed/magnitude binary number as a string. For entering the signed binary number, the user must enter all 32 bits with spaces between for every 4 bits (i.e. 0000 0000 0000 0000 0000 0000 0000 0000). After input, the program passes the string to a function to check if it's valid signed binary number. A valid signed binary number can have 0's, 1's, and spaces. If the string is invalid, then the program will print out their invalid input and print out how the signed binary number should be formatted, then re-prompts the user to enter their signed binary number. If the user enters a valid signed binary number, then the program passes the string to a function that converts the signed binary number to its decimal and returns it. Finally, the program takes the decimal and prints it out using syscall 35 which prints out the integer to binary in two's complement format.

2. MIPS Program implemented in Campos_CSE222_Final_Project.asm file.