EE 231 – Homework 6
Due October 8, 2010

1. Problem 4.16

   Define the carry propagate and carry generate as

$$\begin{aligned} P_i &= A_i + B_i \\ G_i &= A_i B_i \end{aligned}$$

respectively. Show that the output carry and the output sum of a full adder becomes

$$\begin{aligned} C_{i+1} &= (C_i' G_i' + Pi')' \\ S_i &= (P_i G_i') C_i \end{aligned}$$

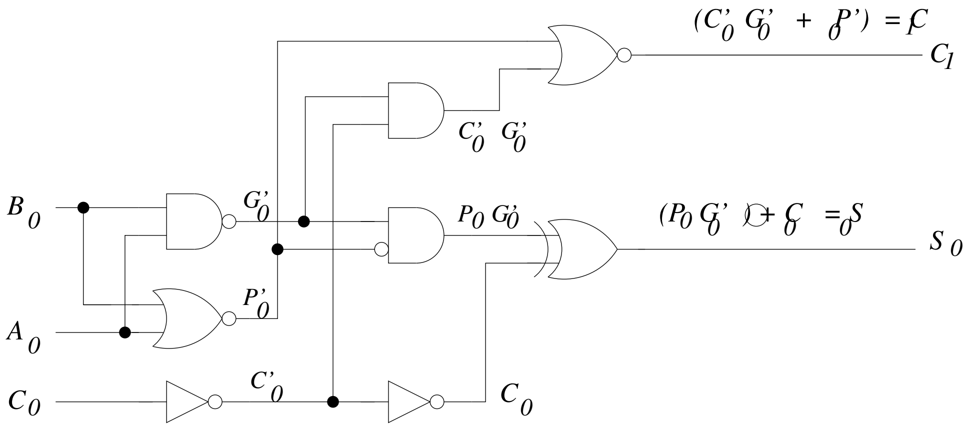
Define $P_i = A_i + B_i$ and $G_i = A_i B_i$. Show that $C_{i+1} = (C_i' G_i' + P_i')'$ and $S_i = (P_i G_i') \oplus C_i$

The output of a full adder is

$$\begin{aligned} S_i &= A_i \oplus B_i \oplus C_i \\ C_{i+1} &= A_i B_i + A_i C_i + B_i C_i \end{aligned}$$

$$\begin{aligned} S_i &= (P_i G_i') \oplus C_i \\ &= [(A_i + B_i)(A_i B_i)'] \oplus C_i \\ &= [(A_i + B_i)(A_i' + B_i')] \oplus C_i \\ &= [A_i A_i' + A_i B_i' + B_i A_i' + B_i B_i'] \oplus C_i \\ &= [A_i B_i' + A_i' B_i] \oplus C_i \\ &= A_i \oplus B_i \oplus C_i \ \text{ QED} \end{aligned}$$

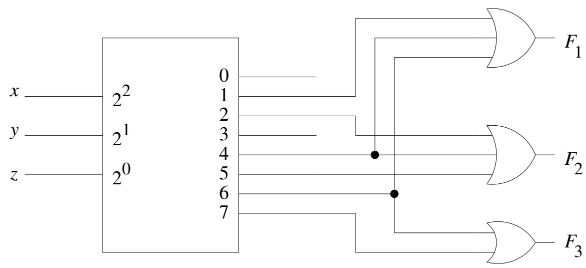
$$\begin{aligned} C_{i+1} &= (C_i' G_i' + P_i')' \\ &= (C_i' G_i')' P_i \\ &= (C_i + G_i) P_i \\ &= (C_i + A_i B_i)(A_i + B_i) \\ &= C_i A_i + C_i B_i + A_i B_i A_i + A_i B_i B_i \\ &= A_i C_i + B_i C_i + A_i B_i + A_i B_i \\ &= A_i C_i + B_i C_i + A_i B_i \ \text{ QED} \end{aligned}$$



2. Using a decoder and external gates, design the combinational circuit defined by the following three Boolean functions:

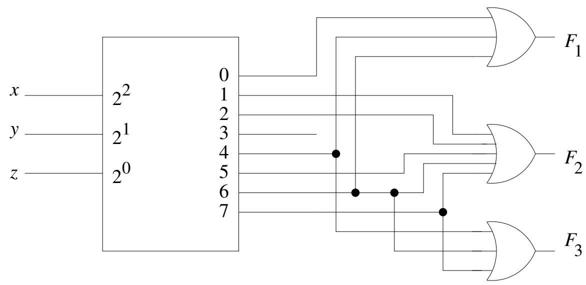
   (a) $F_1 = x'y'z + xz'$
   $F_2 = x'yz' + xy'$
   $F_3 = xyz' + xy$
   Truth table:

x	y	z	F_1	F_2	F_3
0	0	0	0	0	0
0	0	1	1	0	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	1	1	0
1	0	1	0	1	0
1	1	0	1	0	1
1	1	1	0	0	1



   (b) $F_1 = (x + y')z'$
   $F_2 = xz + y'z + yz'$
   $F_3 = (y + z')x$
   Truth table:

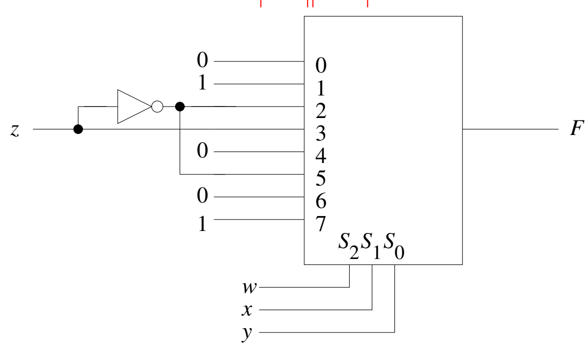
x	y	z	F_1	F_2	F_3
0	0	0	1	0	0
0	0	1	0	1	0
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	1	0	1
1	0	1	0	1	0
1	1	0	1	1	1
1	1	1	0	1	1



3. Implement the following Boolean functions with a multiplexer:

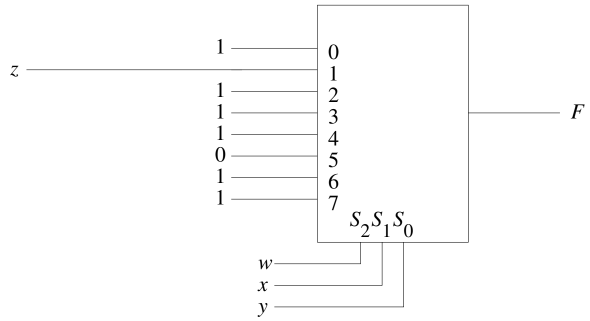
   (a) $F(w, x, y, z) = \Sigma(2, 3, 5, 6, 11, 14, 15)$

w	x	y	z	F	
0	0	0	0	0	$F = 0$
0	0	0	1	0	
0	0	1	0	1	$F = 1$
0	0	1	1	1	
0	1	0	0	0	$F = z'$
0	1	0	1	1	
0	1	1	0	1	$F = z$
0	1	1	1	0	
1	0	0	0	0	$F = 0$
1	0	0	1	0	
1	0	1	0	0	$F = z'$
1	0	1	1	1	
1	1	0	0	0	$F = 0$
1	1	0	1	0	
1	1	1	0	1	$F = 1$
1	1	1	1	1	



(b) $F(w, x, y, z) = \Pi(3, 10, 11)$

w	x	y	z	F	
0	0	0	0	1	$F = 1$
0	0	0	1	1	
0	0	1	0	1	$F = z$
0	0	1	1	0	
0	1	0	0	1	$F = 1$
0	1	0	1	1	
0	1	1	0	1	$F = 1$
0	1	1	1	1	
1	0	0	0	1	$F = 1$
1	0	0	1	1	
1	0	1	0	0	$F = 0$
1	0	1	1	0	
1	1	0	0	1	$F = 1$
1	1	0	1	1	
1	1	1	0	1	$F = 1$
1	1	1	1	1	

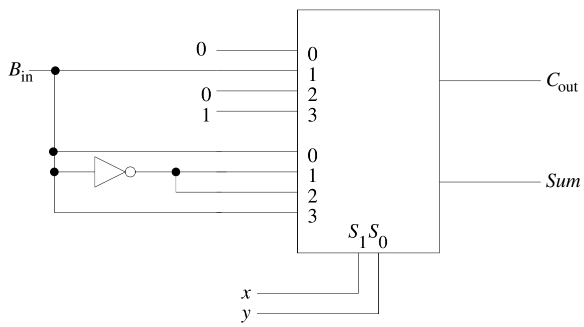


4. Write a Verilog dataflow description to implement the Boolean functions of Problem 3.

```
module hw6_p3(input w, x, y, z, output Fa, Fb);
// Fa = Sum (2,3,5,6,11,14,15)
// Fa = Prod (3,10,11)

// OR together minterms of Fa
assign Fa = (~w & ~x &  y & ~z) |  (~w & ~x &  y &  z) |  (~w &  x & ~y &  z) |
            (~w &  x &  y & ~z) |  ( w & ~x &  y &  z) |  ( w &  x &  y & ~z) |
            ( w &  x &  y &  z);

// AND together maxterms of Fb
assign Fb = (~w | ~x |  y |  z) &  ( w | ~x |  y | ~z) &  ( w | ~x |  y |  z);

endmodule
```

5. Implement a full adder with two 4x1 multiplexers. Note: the truth table for the full adder is:

x	y	C_{in}	C_{out}	Sum
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	1	0
1	1	1	1	1



6. An 8x1 multiplexer has inputs A, B and C connected to the selection inputs S_2, S_1, and S_0 respectively. The data inputs through I_0 through I_7 are as follows:

   (a) $I_1 = I_2 = I_4 = 0; I_3 = I_5 = 1; I_0 = I_7 = D$; and $I_6 = D'$.

   (b) $I_2 = I_3 = 0; I_4 = I_5 = I_7 = 1; I_0 = I_6 = D$; and $I_1 = D'$.

   Determine the Boolean function that the multiplexer implements.

   (a)

   $I_1 = I_2 = I_4 = 0; I_3 = I_5 = 1; I_0 = I_7 = D$; and $I_6 = D'$.

A	B	C	D	F	
0	0	0	0	1	$F = D$
0	0	0	1	0	
0	0	1	0	0	$F = 0$
0	0	1	1	0	
0	1	0	0	0	$F = 0$
0	1	0	1	0	
0	1	1	0	1	$F = 1$
0	1	1	1	1	
1	0	0	0	0	$F = 0$
1	0	0	1	0	
1	0	1	0	1	$F = 1$
1	0	1	1	1	
1	1	0	0	0	$F = D'$
1	1	0	1	1	
1	1	1	0	1	$F = D$
1	1	1	1	0	

   $F = A'B'C'D' + A'BCD' + A'BCD + AB'CD' + AB'CD + ABC'D + ABCD'$
   $F = A'B'C'D' + A'BC + AB'C + ABC'D + ACD'$

(b)

$I_2 = I_3 = 0; I_4 = I_5 = I_7 = 1; I_0 = I_6 = D;$ and $I_1 = D'$.

A	B	C	D	F	
0	0	0	0	1	$F = D$
0	0	0	1	0	
0	0	1	0	0	$F = D'$
0	0	1	1	1	
0	1	0	0	0	$F = 0$
0	1	0	1	0	
0	1	1	0	0	$F = 0$
0	1	1	1	0	
1	0	0	0	1	$F = 1$
1	0	0	1	1	
1	0	1	0	1	$F = 1$
1	0	1	1	1	
1	1	0	0	1	$F = D$
1	1	0	1	0	
1	1	1	0	1	$F = 1$
1	1	1	1	1	

$F = A'B'C'D' + A'B'CD + AB'C'D' + AB'C'D + AB'CD' + AB'CD + ABC'D' + ABCD$
$F = B'C'D' + B'CD + AB' + AD' + AC$

7. Problem 4.39. Use a behavioral description to implement the problem. Do not write a gate level or dataflow description.

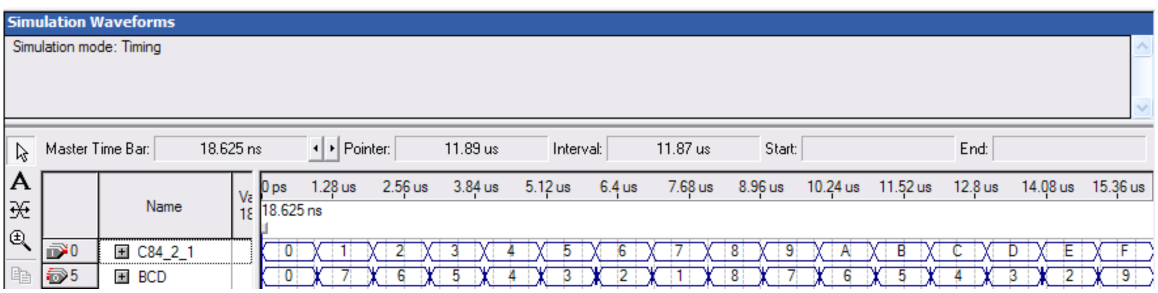
   Write an HDL behavioral description of a four-bit comparator with a six-bit output $Y[5:0]$. Bit 5 of Y is for "equals", bit 4 is for "not equal to", bit 3 is for "greater than", bit 2 is for "less than", bit 1 for "greater than or equal to", and bit 0 for "less than or equal to".

```
module four_bit_comparator(input [3:0] A, B, output reg [5:0] Y);

always @(A, B) begin
Y = 6'b000000;
if (A == B) Y[5] = 1'b1;
if (A != B) Y[4] = 1'b1;
if (A >  B) Y[3] = 1'b1;
if (A >= B) Y[2] = 1'b1;
if (A <  B) Y[1] = 1'b1;
if (A <= B) Y[0] = 1'b1;
end

endmodule
```

8. Problem 4.50.

   Using a case statement, develop and simulate a behavioral model of the 84-2-1 to BCD code converter described in Problem 4.8.

```
module code_converter(input [3:0] C84_2_1, output reg [3:0] BCD);

always @(C84_2_1)
    case (C84_2_1)
        4'b0000:  BCD = 4'b0000;
        4'b0111:  BCD = 4'b0001;
        4'b0110:  BCD = 4'b0010;
        4'b0101:  BCD = 4'b0011;
        4'b0100:  BCD = 4'b0100;
        4'b1011:  BCD = 4'b0101;
        4'b1010:  BCD = 4'b0110;
        4'b1001:  BCD = 4'b0111;
        4'b1000:  BCD = 4'b1000;
        4'b1111:  BCD = 4'b1001;
default:  BCD = 4'bxxxx;
endcase
endmodule
```



9. Using a case statement, write an HDL behavioral description of an eight-bit arithmetic-logic unit (ALU). The ALU needs to implement the 10 functions listed below. The inputs are two eight-bit numbers A and B, and select inputs S (where S has enough bits to select the ten functions). The outputs are the eight-bit result R, a zero-bit Z, and a carry bit C. The C bit is described in the table below. (X means Don't Care.) The zero bit Z is 1 if all the bits of the eight-bit result are 0, and is 0 otherwise.

Name	Description	R	C	Z
LOAD	Load input A	A	X	1 if $R == 0$
ADDA	Add inputs	$A + B$	Carry	1 if $R == 0$
SUBA	Subtract inputs	$A - B$	Borrow	1 if $R == 0$
ANDA	AND inputs	$A\&B$	X	1 if $R == 0$
ORAA	OR inputs	$A\|B$	X	1 if $R == 0$
COMA	Bitwise Complement input A	$\sim A$	1	1 if $R == 0$
INCA	Increment input A	$A + 1$	X	1 if $R == 0$
LSRA	Logical Shift Right input A	$0 => R[7]$ $A[7:1] => R[6:0]$	$A[0]$	1 if $R == 0$
LSLA	Logical Shift Left input A	$0 => R[0]$ $A[6:0] => R[7:1]$	$A[7]$	1 if $R == 0$
ASRA	Arithmetic Shift Right input A	$A[7] => R[7]$ $A[7:1] => R[6:0]$	$A[0]$	1 if $R == 0$