HMK #6 ECE 331

(2)(a) XOR Truth Table
$a \oplus b = c$

a	b	c
0	0	0
0	1	1
1	0	1
1	1	0

Inputs:

-S	$-A_3$	A_2	A_1	A_0	$-B_3$	B_2	B_1	B_0
0	1	0	1	1	0	1	1	1

First Adder:

$$\begin{array}{r} B_0 \oplus S \\ + \quad A_0 \\ \hline (S)\ C_0 \end{array} \Rightarrow \begin{array}{r} 1 \\ + \ 1 \\ \hline 0 \end{array}$$

$(C_2)\,1\,0\,(S_0)$ — C_1 ↗ ↖ S_0

Second Adder:

$$\begin{array}{r} B_1 \oplus S \\ + \quad A_1 \\ \hline C_1 \end{array} \Rightarrow \begin{array}{r} 1 \\ + \ 1 \\ \hline 1 \end{array}$$

$C_2 \rightarrow 1\,1 \leftarrow S_1$

Third Adder:

$$\begin{array}{r} B_2 \oplus S \\ + \quad A_2 \\ \hline C_2 \end{array} \Rightarrow \begin{array}{r} 1 \\ + \ 0 \\ \hline 1 \end{array}$$

$C_3 \rightarrow 1\,0 \leftarrow S_2$

Fourth Adder:

$$\begin{array}{r} B_3 \oplus S \\ + \quad A_3 \\ \hline C_3 \end{array} \Rightarrow \begin{array}{r} 0 \\ + \ 1 \\ \hline 1 \end{array}$$

$C_4 \rightarrow 1\,0 \leftarrow S_3$

C_4	S_3	S_2	S_1	S_0
1	0	0	1	0

Inputs:

-S	$-A_3$	A_2	A_1	A_0	$-B_3$	B_2	B_1	B_0
1	1	1	0	0	0	0	1	1

First Adder:

$B_0 \oplus S$ (1 ⊕ 1) = 0

$+ \quad A_0 \Rightarrow + 0$

$(S)\, C_0 \qquad\quad 1$

$C_1 \rightarrow 0\,1 \leftarrow S_0$

Second Adder:

$B_1 \oplus S$ (1 ⊕ 1) = 0

$+ \quad A_1 \Rightarrow + 0$

$\quad C_1 \qquad\quad 0$

$C_2 \rightarrow 0\,0 \leftarrow S_1$

Third Adder:

$B_2 \oplus S$ (0 ⊕ 1) = 1

$+ \quad A_2 \Rightarrow 1$

$\quad C_2 \qquad\quad 0$

$C_3 \rightarrow 0\,1 \leftarrow S_2$ (carry 1, 0)

Forrth Adder:

$B_3 \oplus S$ (0 ⊕ 1) = 1

$+ \quad A_3 \Rightarrow + 1$

$\quad C_3 \qquad\quad 0\,1$

$C_4 \rightarrow 1\,0 \leftarrow S_3$ (carry 1)

C_4	S_3	S_2	S_1	S_0
1	0	1	0	1

(carry 1, 0)

So $A_3 - A_0 = 12$

$B_3 - B_0 = -3$

Result: 9

Inputs:

-S	$-A_3$	A_2	A_1	A_0	$-B_3$	B_2	B_1	B_0
1	0	1	0	1	1	0	0	0

First Adder: $1 + 1 + 1 = 11$ ← S_0 (carry C_1)

Second Adder: $1 + 0 + 1 = 10$ ← S_1 (carry C_2)

Third Adder: $1 + 1 + 1 = 11$ ← S_2 (carry C_3)

Fourth Adder: $0 + 0 + 1 = 01$ ← S_3 (carry C_4)

C_4	S_3	S_2	S_1	S_0
0	1	1	0	1

NOTE: Logic is exactly the same as in the last two problems, the only thing that change are the values.

Inputs:

-S	$-A_3$	A_2	A_1	A_0	$-B_3$	B_2	B_1	B_0
0	0	1	0	0	1	1	0	0

First Adder: $0 + 0 + 0 = 00$ ← S_0 (carry C_1)

Second Adder: $0 + 0 + 0 = 00$ ← S_1 (carry C_2)

Third Adder: $1 + 1 + 0 = 10$ (right) ← S_2 (carry C_3)

Fourth Adder: $1 + 0 + 1 = 10$ right ← S_3 (carry C_4)

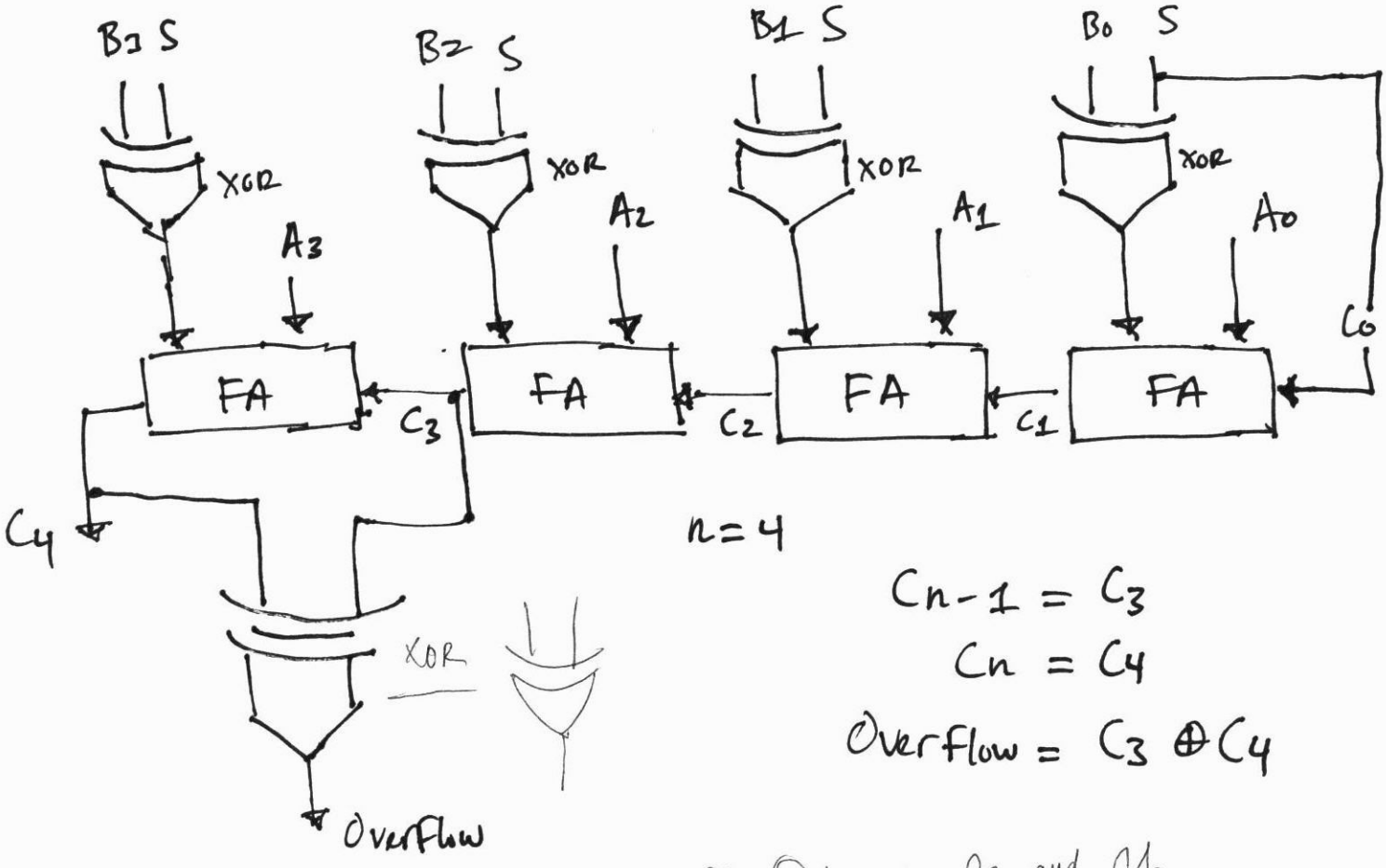
C_4	S_3	S_2	S_1	S_0
1	0	0	0	0

is the result (10000)

2 b)

Add/Substractor Unit with Overflow Detection

To detect OverFlow when adding/substracting two n-bit numbers, we just need to check the output from the function: $C_{n-1} \oplus C_n$. IF the output is high, there is an overflow otherwise it is not. The circuit from problem 1 can be modified to detect overflow as follows:



n=4

$$C_{n-1} = C_3$$

$$C_n = C_4$$

$$\text{Overflow} = C_3 \oplus C_4$$

Obs. ① INPUTS: C_3 and C_4

OUTPUT: OVERFLOW

② Truth table

C_3	C_4	OVERFLOW
0	0	0
0	1	1
1	0	1
1	1	0

3a

a) Truth Table for given circuit

X	Y	Z	M	G
0	0	0	(X) 0	0 (Y)
0	0	1	(Y) 0	0 (Y)
0	1	0	(X) 0	1 (Y)
0	1	1	(Y) 1	0 ('0')
1	0	0	(X) 1	0 ('0')
1	0	1	(Y) 0	0 (Y)
1	1	0	(X) 1	0 ('0')
1	1	1	(Y) 1	0 ('0')

For M:

For $z=0 \rightarrow M=X$

For $z=1 \rightarrow M=Y$

For G:

For $M=0 \rightarrow G=Y$

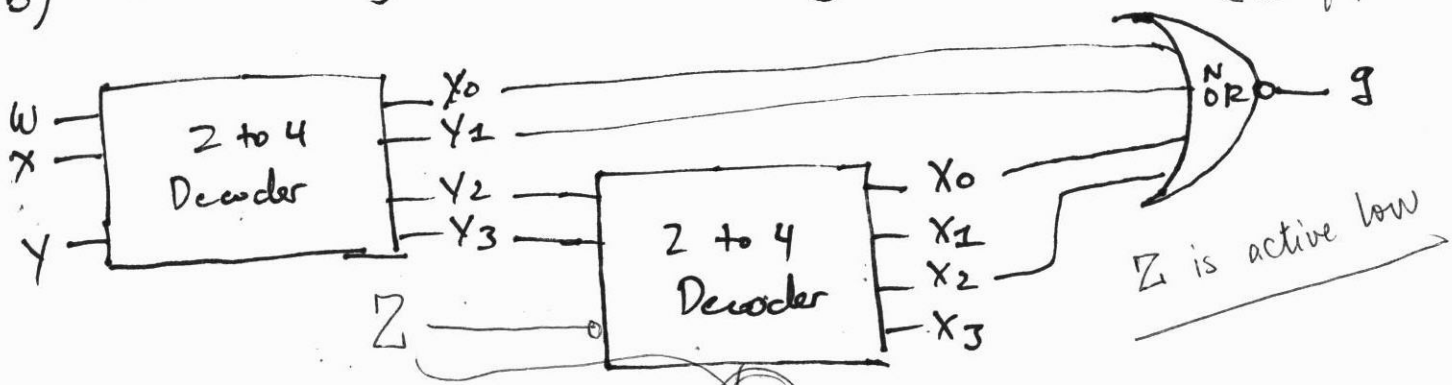
For $M=1 \rightarrow G='0'$

Minimized Boolean Equation: $g = \bar{X}\,Y\,\bar{Z}\,M$

M is not an input in this circuit! Careful.

3b

b) After Adding intermediate signals:



W	X	Y	Y_0	Y_1	Y_2	Y_3	Z	X_0	X_1	X_2	X_3	g
0	0	0	0	0	0	0	1	0	0	0	0	1
0	0	1	1	0	0	0	0	1	0	0	0	0
0	1	0	0	0	0	0	1	0	0	0	0	1
0	1	1	0	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0	0	1
1	0	1	0	0	1	0	0	0	0	1	0	0
1	1	0	0	0	0	0	1	0	0	0	0	1
1	1	1	0	0	0	1	0	0	1	0	0	1

$g = \bar{w}\bar{x}\bar{y}\bar{z} + \bar{w}x\bar{y}\bar{z} + w\bar{x}\bar{y}\bar{z} + wx\bar{y}\bar{z} + wxyz$

$g = \bar{y}\bar{z} + wxyz$

~~(3)~~ (4) A Demultiplexer is nothing but the opposite of a MUX. where a MUX maps many inputs to one output, a demux places the value of a single data input onto many data outputs. Any n to 2^n decoder, (n = # of bits) performs this function and can be implemented as a demultiplexer.

Circuit shown Below: 1 to 4 Demux with Enable (2 to 4 Decoder)

① step INPUTS: En W_0 W_1 OUTPUTS: Y_0 Y_1 Y_2 Y_3

2. step: truth table.

3. step: minimization

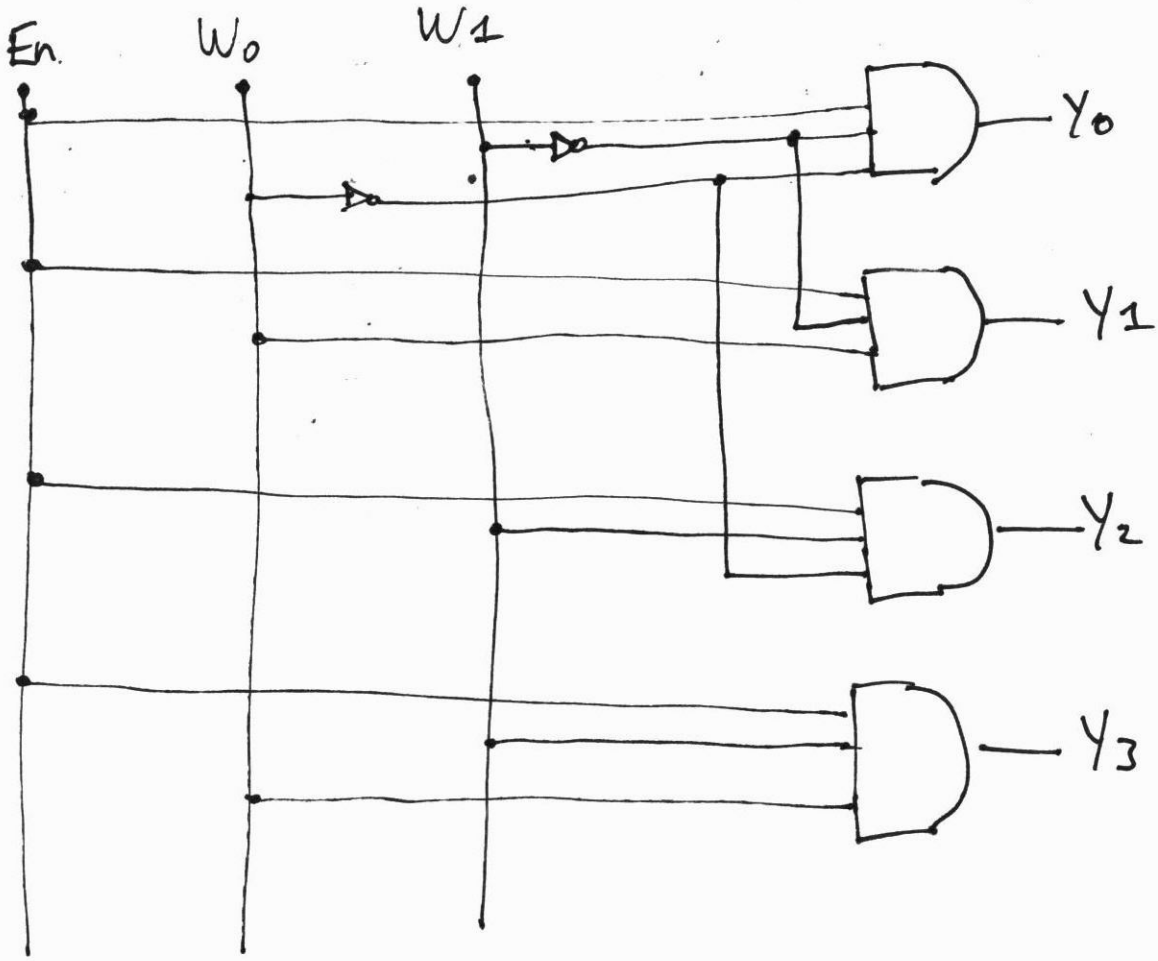
4 step - circuit implementation

$Y_0 = En\,\bar{W}_1\bar{W}_0$

$Y_1 = En\,\bar{W}_1 W_0$

$Y_2 = En\,W_1\bar{W}_0$

$Y_3 = En\,W_1 W_0$



this is not asked in question

Is this question 4 ? If so, this is not the way the question asks

(5) VHDL CODE

- Question does not specify that the demux will have an enable. A very simple way to implement a demux without an enable is the following:

```
Library ieee;
use ieee. std_logic_1164.all

Entity one to eight_ demux is
  port ( x: in std_logic_vector (2 downto 0);
         y: out std_logic_vector (7 downto 0) );
 end     one to eight_demux;
Architecture demux_arch of one to eight_ demux is
 y <= "0000 0001" when x="000" else
 y <= "0000 0010" when x="001" else
 y <= "00000100" when x="010" else
 y <= "0000 1000" when x="011" else
 y <= "00010000" when x="100" else
 y <= "0010 0000" when x="101" else
 y <= "0100 0000" when x="110" else
 y <= "1000 0000" when x="111";
end demux_arch;
```

Notice that 'x' vector drives what output-bit should go high on the output vector 'y'.

Example -> x="010"
y(2) = should go high

So output vector 'y' will look like: y="0000 0010".

EXTRA CREDIT

	74HCT42	SN74LVC168	47HC4511
1. LOW SIGNALS	1	0	0
2. ONLY DEMUX	0	1	0
3. ONLY DECODER	0	0	1
4. DECODER OR DEMUX	1	0	1
5. COSTS MORE THAN DOLLAR	~~1~~ 0	~~1~~ 0	~~1~~ 0

I think this can be "1"

[These (all) cost less than $1,-
To check that: go to
www.digikey.com,
type the IC numbers in, etc.]