CprE 281: Digital Logic
Midterm 2: Monday Oct. 28, 2013

Student Name: **Student ID Number:**

Lab Section: Mon 9-12(N), Tue 2-5(M), Wed 8-11(J), Thu 2-5(L), Thu 5-8(K), Fri 11-2(G)
(circle one)

1. True/False Questions (10 x 1p each = 10p)

(a) I forgot to write down my name and student ID number. TRUE / [✓ FALSE]

(b) An edge-triggered D flip-flop can be implemented with 6 NAND gates. [✓ TRUE] / FALSE

(c) An XOR gate can be implemented with 2 NAND gates. TRUE / [✓ FALSE]

(d) Any Boolean function can be implemented using only 2-to-1 multiplexers. [✓ TRUE] / FALSE

(e) Any Boolean function can be implemented using only AND and OR gates. TRUE / [✓ FALSE]

(f) Any Boolean function can be implemented using only XNOR gates. TRUE / [✓ FALSE]

(g) The outputs of a binary encoder are one-hot encoded. TRUE / [✓ FALSE]

(h) The outputs of a priority encoder are one-hot encoded. TRUE / [✓ FALSE]

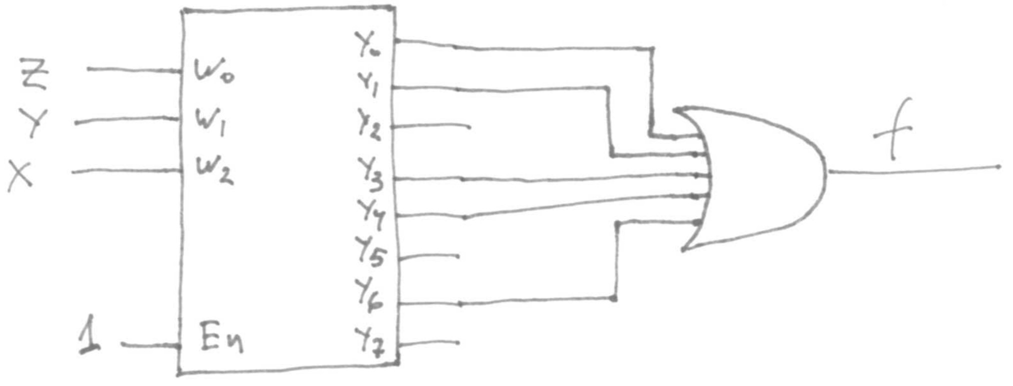
(i) Binary subtraction is easier with 2's complement than with 1's complement. [✓ TRUE] / FALSE

(j) to_be | ~ to_be [✓ TRUE] / FALSE

2. Function Implementation with a Decoder (10p)

Implement the Boolean function $f(x, y, z) = \Pi M(2,5,7)$ using a 3-to-8 decoder and one OR gate. Draw the circuit diagram and clearly label all inputs, pins, and outputs.

To use the decoder we need the minterms, not the maxterms.
In this case, the equivalent function is: $f(X,Y,Z) = \Sigma m(0,1,3,4,6)$.



3.Binary Addition and Subtraction (5 x 3p each = 15p)

Convert the following integers into binary numbers and perform the addition or subtraction using 2's complement if necessary. Write your answers and all intermediary steps to the right of each problem. Use 5-bit numbers for all problems and indicate if any bits need to be ignored.

$$\begin{array}{rr} & (+4) \\ + & (+3) \\ \hline & (+7) \end{array} \qquad \begin{array}{rr} & 00100 \\ + & 00011 \\ \hline & 00111 \end{array}$$

$$\begin{array}{rr} & (+5) \\ + & (-3) \\ \hline & (+2) \end{array} \qquad \begin{array}{rr} & 00101 \\ + & 11101 \\ \hline \textcircled{1} & 00010 \end{array}$$

Ignore

convert to 2's complement

$$\begin{array}{rr} & (-6) \\ - & (-3) \\ \hline & (-3) \end{array} \qquad \begin{array}{rr} & (-00110) \\ - & (-00011) \\ \hline \end{array} \begin{array}{c} \Rightarrow \\ \Rightarrow \end{array} \begin{array}{rr} & 11010 \\ - & 11101 \\ \hline \end{array} \Rightarrow \begin{array}{r} 11010 \\ 00011 \\ \hline 11101 \end{array}$$

$$\begin{array}{rr} & (-12) \\ + & (+10) \\ \hline & (-2) \end{array} \qquad \begin{array}{rr} & (-01100) \\ + & (+01010) \\ \hline \end{array} \Rightarrow \begin{array}{rr} & 10100 \\ + & 01010 \\ \hline & 11110 \end{array}$$

$$\begin{array}{rr} & (+14) \\ + & (-9) \\ \hline & (+5) \end{array} \qquad \begin{array}{rr} & (+01110) \\ + & (-01001) \\ \hline \end{array} \Rightarrow \begin{array}{rr} & 01110 \\ + & 10111 \\ \hline \textcircled{1} & 00101 \end{array}$$

Ignore

4. Number Conversions (4 x 5p each = 20p)

(a) Convert $\mathbf{BF400000_{16}}$ (a 32-bit float stored in IEEE 754 format) to decimal:

$$\underset{\text{negative}}{1}\;|\;\underbrace{0111\;1111\;0}_{126_{10}}\;|\;\overset{=\frac{1}{2}}{1}00\;0000\;0000\;0000\;0000\;0000$$

$$(-1)^1 \times 2^{126-127} \times \left(1+\frac{1}{2}\right) = -1 \times 2^{-1} \times 1.5 = -\frac{1.5}{2} = -0.75$$

(b) Convert the following 32-bit float number (in IEEE 754 format) to decimal

$$\underset{\text{negative}}{\mathbf{1}}\;|\;\underbrace{\mathbf{1\;0\;0\;0\;0\;0\;1\;1}}_{131_{10}}\;|\;\overset{2^{-1}}{\mathbf{0}}\;\overset{2^{-2}}{\mathbf{0}}\;\overset{2^{-3}}{\underset{=\frac{1}{8}}{\mathbf{1}}}\;\mathbf{0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0\;0}$$

$$(-1)^1 \times 2^{131-127} \times \left(1+\frac{1}{8}\right) = -1 \times 2^4 \times \left(\frac{8+1}{8}\right) =$$

$$= -16 \times \frac{9}{8} = -18$$

(c) Write down the 32-bit floating point representation for the real number 10.0

$$10 / 8 = 1.25 \quad \Rightarrow \quad 10.0 = (-1)^0 \times 2^3 \times 1.25$$

$$= (-1)^0 \times 2^{130-127} \times \left(1+\frac{1}{4}\right)$$

```
  10
-  8
  20
- 16
  40
- 40
   0
```

Result:

$$\underset{\text{positive}}{0}\;|\;\underbrace{10000010}_{130_{10}}\;|\;01\underbrace{00\ldots\;0}_{21\text{ zeros}}\;|$$

(d) Write down the 32-bit floating point representation for the real number -5.5

$$-5.5 / 4 = -1.375 \quad \Rightarrow \quad -5.5 = -1 \times 2^2 \times 1.375$$

$$= (-1)^1 \times 2^{129-127} \times (1+0.375)$$

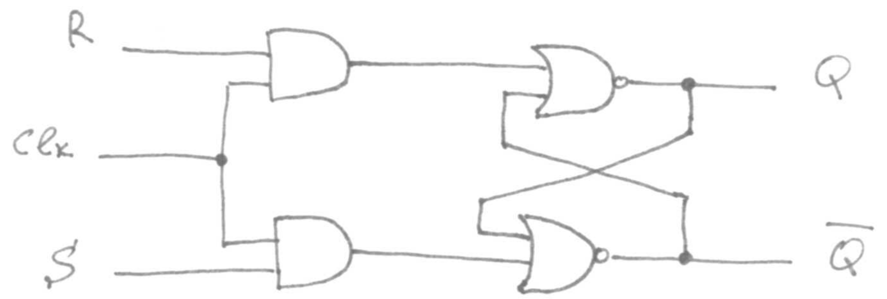
```
  4
 15
- 12
  30
- 28
  20
- 20
   0
```

$$0.375 \times 2 = 0.750$$
$$0.75 \times 2 = 1.50$$
$$0.5 \times 2 = 1.0$$
$$0.0 \times 2 \quad 0.0$$

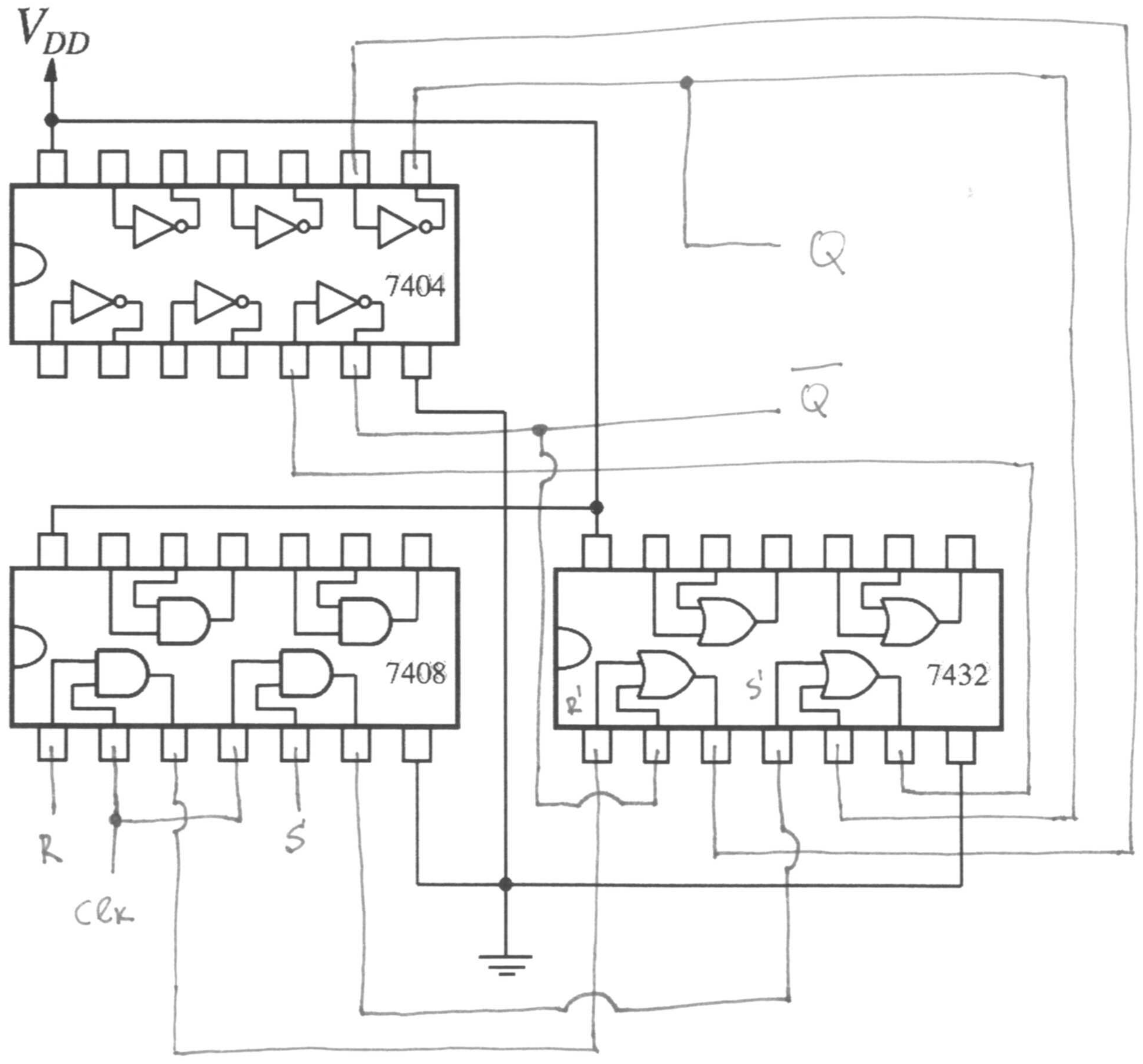
result: $1\,|\,10000001\,|\,01100\ldots\ldots\;0\,|$

5. Chip Implementation (10p)

a) Draw the circuit diagram for a gated SR latch.

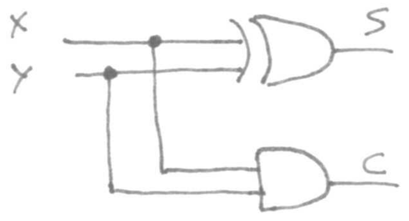


b) Implement the gated SR latch from a) using only the available chips shown below. Add the necessary wires and labels for the inputs and outputs to get the desired circuit.



6. Half-Adder with 2-to-1 Multiplexers (10p)

Implement a half-adder using only 2-to-1 multiplexers and no other logic gates.
Add the necessary wires and label clearly all inputs and outputs of your circuit.
Show your intermediary calculations and derivations and clearly indicate your final result.

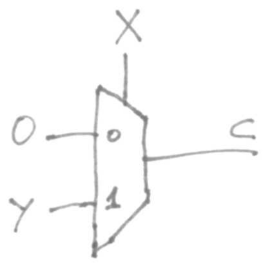


X	Y	Sum S	Carry C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Let's derive c first:

X	Y	C
0	0	0
0	1	0
1	0	0
1	1	1

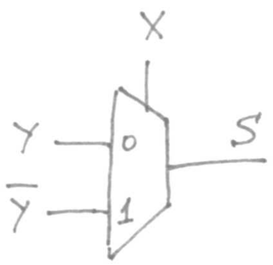
X	C
0	0
1	Y



Derivation for S:

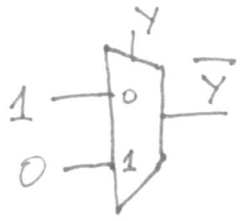
X	Y	S
0	0	0
0	1	1
1	0	1
1	1	0

X	S
0	Y
1	$\overline{Y}$

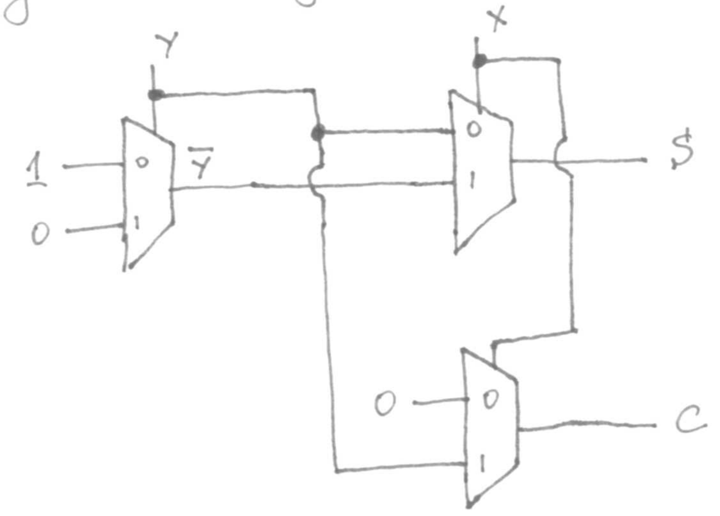


We also need to derive $\overline{Y}$:

Y	$\overline{Y}$
0	1
1	0



Putting it all together:



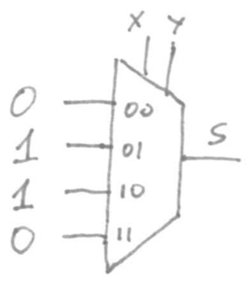
7. Half-Adder with 4-to-1 Multiplexers (10p)

Implement a half-adder using only 4-to-1 multiplexers and <u>no additional</u> logic gates. Add the necessary wires and label clearly all inputs and outputs of your circuit. Show your intermediary calculations and derivations and clearly indicate your final result.

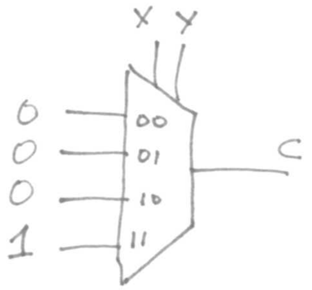
Truth table for the half-adder:

X	Y	S	C
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

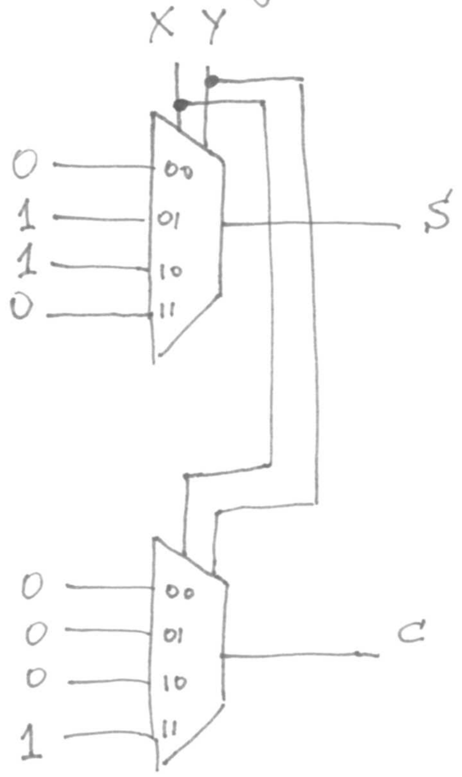
Derivation for S:



Derivation for C:



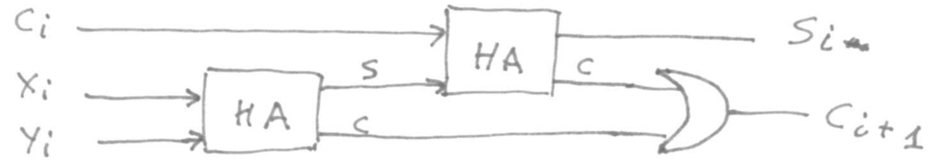
Putting it all together:



8. Full-Adder with 4-to-1 Multiplexers (10p)

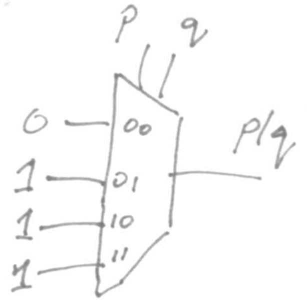
Implement a full-adder using only 4-to-1 multiplexers. No other logic gates are allowed. Add the necessary wires and label all inputs and outputs of your circuit. Show your intermediary calculations and derivations and clearly indicate your final result.

A full-adder can be implemented with two half-adders and one OR gate. The high-level circuit diagram is:

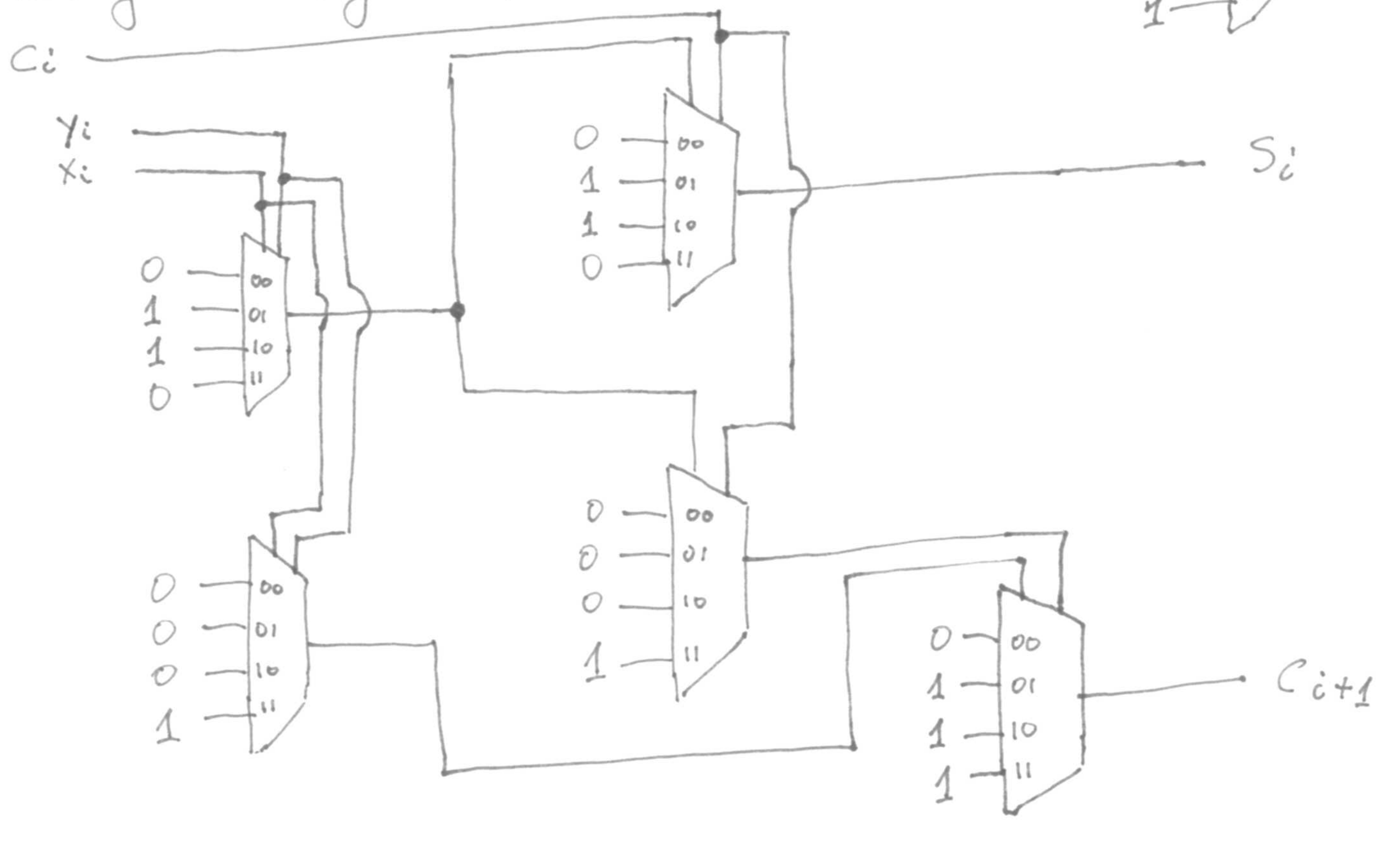


From problem 7 we already know how to implement a half-adder using only 4-to-1 multiplexers. The OR gate can be implemented as follows:

p	q	p|q
0	0	0
0	1	1
1	0	1
1	1	1

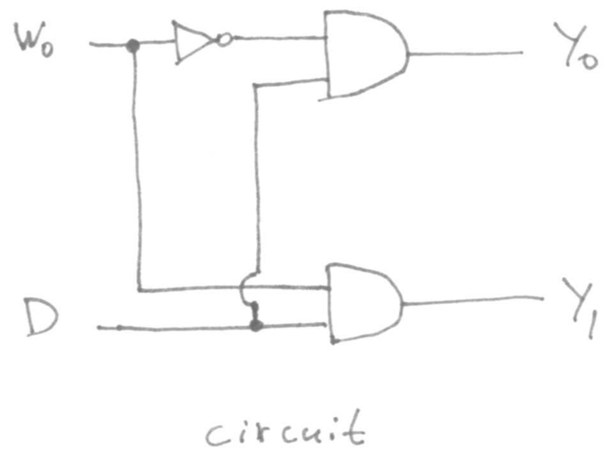


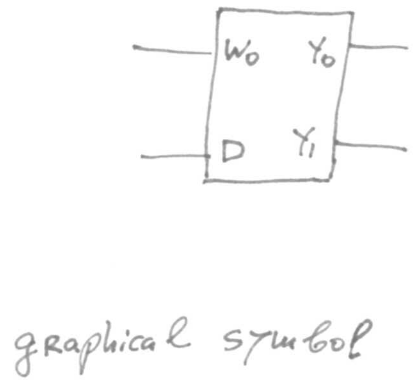
Putting it all together:



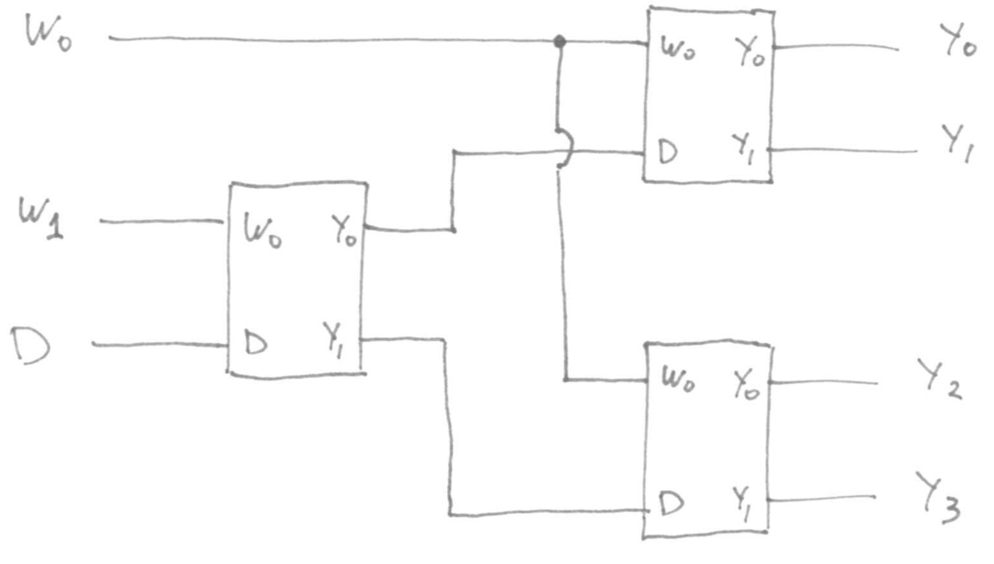
9. Demultiplexers (5p + 10p = 15p)

(a) Draw the circuit diagram for a 1-to-2 demultiplexer. Label all inputs and outputs.





(a) Use several instances of your circuit from part (a) to build a 1-to-4 demultiplexer.
Hint: come up with a graphical symbol for your solution in part (a) to simplify things.



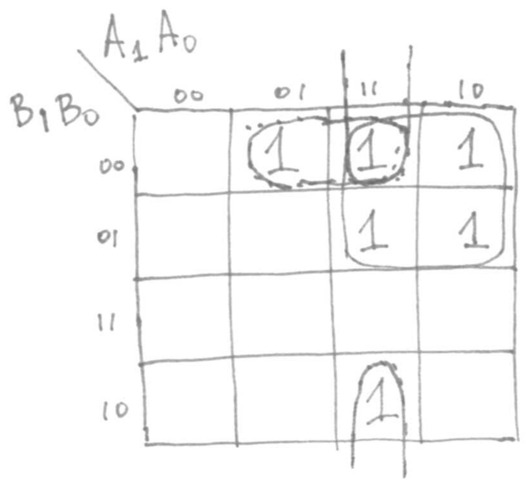
10. Comparator Circuit (4 x 5p = 20p) [Use the space on the next page if needed.]

a) Draw the combined truth table for a function with two outputs that compares two 2-bit binary numbers (let's call them A and B). The first output is 1 if A > B and 0 otherwise. The second output is 1 if A<B and zero otherwise.
b) Use a K-map to optimize the first output.
c) Use a K-map to optimize the second output.
d) Draw the circuit diagram for the two-bit comparator circuit.

a)

A_1	A_0	B_1	B_0	$A > B$	$A < B$
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	0	1
0	0	1	1	0	1
0	1	0	0	1	0
0	1	0	1	0	0
0	1	1	0	0	1
0	1	1	1	0	1
1	0	0	0	1	0
1	0	0	1	1	0
1	0	1	0	0	0
1	0	1	1	0	1
1	1	0	0	1	0
1	1	0	1	1	0
1	1	1	0	1	0
1	1	1	1	0	0

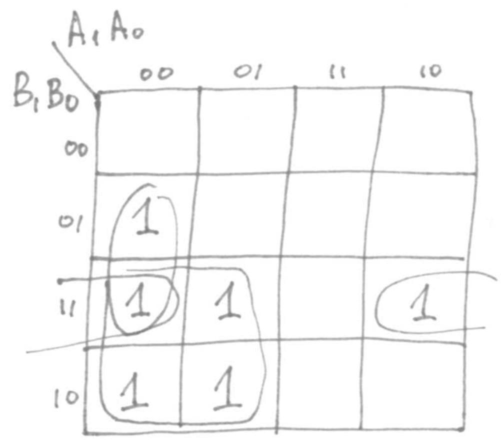
b)



B_1B_0 \ A_1A_0	00	01	11	10
00		1	1	1
01			1	1
11				
10			1	

$$f_>(A_1,A_0,B_1,B_0) = A_1\overline{B_1} + A_1A_0\overline{B_0} + A_0\overline{B_1}\,\overline{B_0}$$

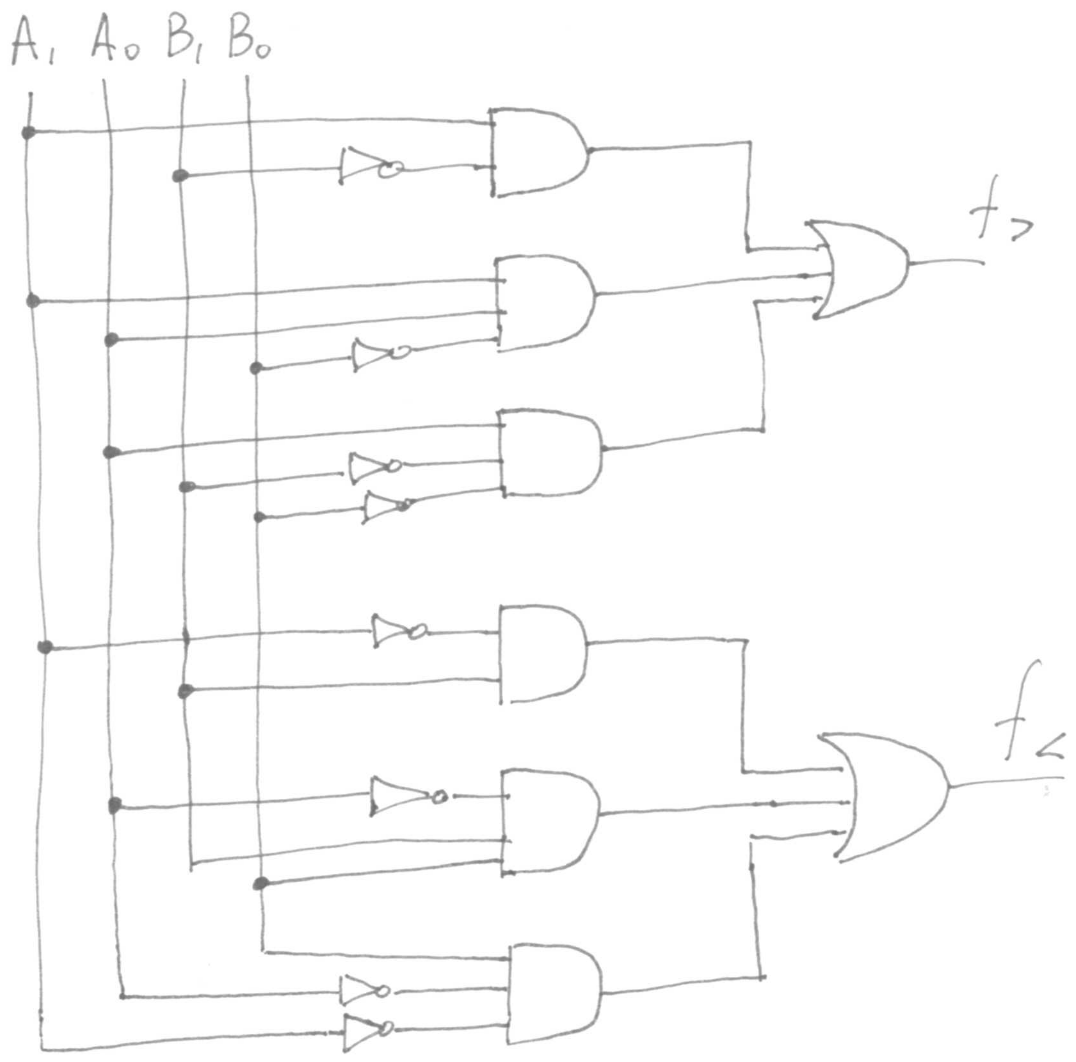
c)



B_1B_0 \ A_1A_0	00	01	11	10
00				
01	1			
11	1	1		1
10	1	1		

$$f_<(A_1,A_0,B_1,B_0) = \overline{A_1}B_1 + \overline{A_0}B_1B_0 + \overline{A_1}\,\overline{A_0}B_0$$

d)



Question	Max	Score
1. True/False	10	
2. Decoders	10	
3. Addition/Subtraction	15	
4. Number Conversions	20	
5. Chip Implementation	10	
6. Half-Adder (2-to-1)	10	
7. Half-Adder (4-to-1)	10	
8. Full-Adder (4-to-1)	10	
9. Demultiplexers	15	
10. Comparator Circuit	20	
TOTAL:	130	