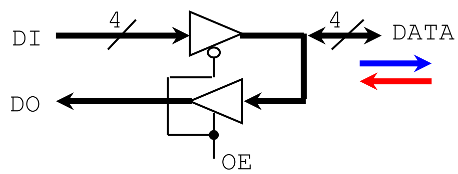
Solutions - Homework 2

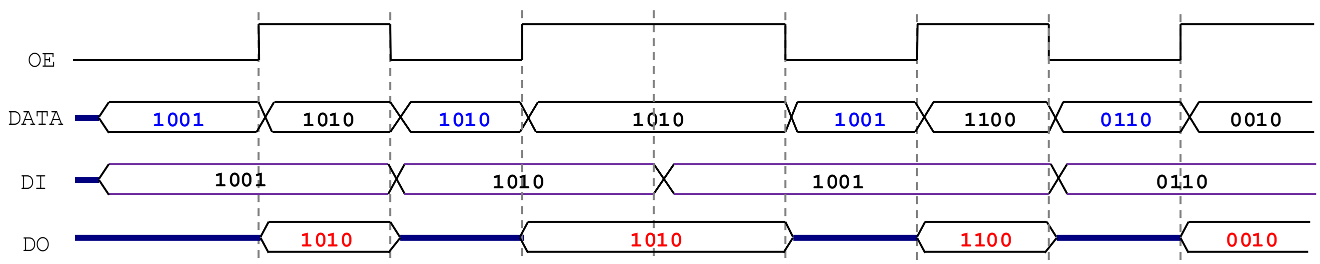
(Due date: February 4th @ 5:30 pm)

Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (5 PTS)

- For the following 4-bit bidirectional port, complete the timing diagram (signals DO and $DATA$):





PROBLEM 2 (15 PTS)

a) What is the minimum number of bits required to represent: (2 pts)

✓ 220,000 colors?
$\lceil \log_2 220{,}000 \rceil = 18$

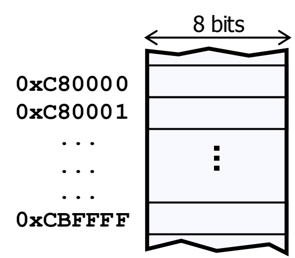
✓ Numbers between 65,000 and 69,096?
$\lceil \log_2 (69096 - 65000 + 1) \rceil = \lceil \log_2 4097 \rceil = 13$

b) A microprocessor has a 24-bit address line. The size of the memory contents of each address is 8 bits. The memory space is defined as the collection of memory positions the processor can address. (5 pts)

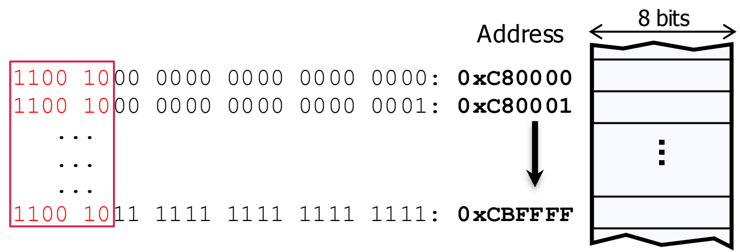
- What is the address range (lowest to highest, in hexadecimal) of the memory space for this microprocessor? What is the size (in bytes, KB, or MB) of the memory space? 1KB = 2^{10} bytes, 1MB = 2^{20} bytes, 1GB = 2^{30} bytes

  Address range: 0x000000 to 0xFFFFFF
  With 24 bits, we can address 2^{24} bytes, thus we have $2^4 2^{20}$ = 16 MB

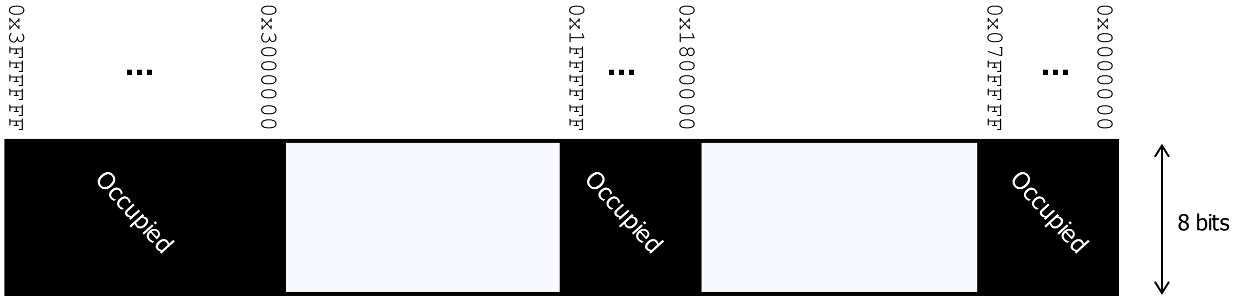
- A memory device is connected to the microprocessor. Based on the size of the memory, the microprocessor has assigned the addresses 0xC80000 to 0xCBFFFF to this memory device.
  - What is the size (in bytes, KB, or MB) of this memory device?
  - What is the minimum number of bits required to represent the addresses only for this memory device?



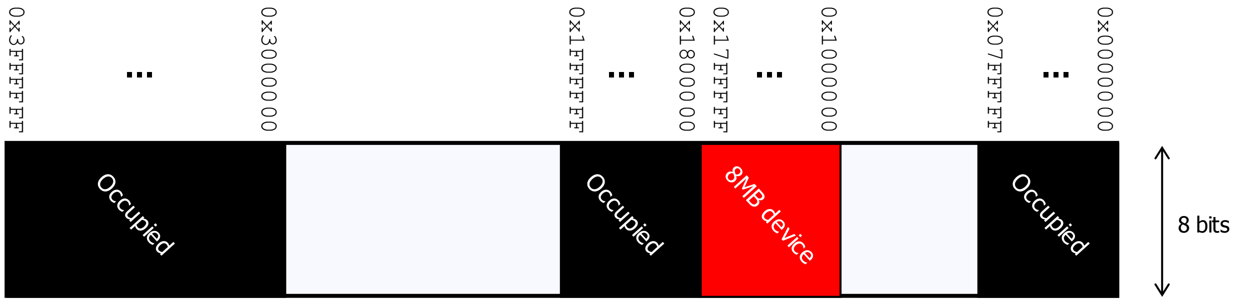
As per the figure, we only need 18 bits for the addresses in the given range. Thus, the size of the memory device is 2^{18} = 256 KB.



c) The figure below depicts the entire memory space of a microprocessor. Each memory address occupies one byte. (8 pts)
- What is the size (in bytes, KB, or MB) of the memory space? What is the address bus size of the microprocessor?
- If we have a memory chip of 8MB, how many bits do we require to address 8MB of memory?
- We want to connect the 8MB memory chip to the microprocessor. Provide an address range so that 8MB of memory is properly addressed. You can only use any of the non-occupied portions of the memory space as shown below.



- Address Range: `0x0000000` to `0x3FFFFFF`. To represent all these addresses, we require 26 bits. So, the address bus size of the microprocessor is 26 bits. The size of the memory space is then 2^{26} = 64MB.
- 8MB memory device: 8MB = $2^3 2^{20}$ = 2^{23} bytes. Thus, we require 23 bits to address the memory device.
  The 23-bit address range for an 8MB memory would be: `0x000000` to `0x7FFFFF`. To place this range within the 26-bit memory space in the figure, we have four options. We selected: `0x1000000` to `0x17FFFFF`.



PROBLEM 3 (20 PTS)

- In these problems, you MUST show your conversion procedure. **No procedure ≡ zero points**.

a) Convert the following decimal numbers to their 2's complement representations: binary and hexadecimal. (6 pts)

✓ -93.3125, 172.65625, -64.5078125, -71.25.

- 93.3125 = 01011101.0101 → -93.3125 = 10100010.1011 = 0xA2.B
- 172.65625 = 010101100.10101 = 0x0AC.A8
- 64.5078125 = 01000000.1000001 → -64.5078125 = 10111111.0111111 = 0xBF.7E
- 71.25 = 01000111.01 → -71.25 = 10111000.11

b) Complete the following table. The decimal numbers are unsigned: (8 pts.)

Decimal	BCD	Binary	Reflective Gray Code
299	001010011001	100101011	110111110
587	010110000111	1001001011	1101101110
1587	0001010110000111	11000110011	10100101010
128	000100101000	10000000	11000000
166	000101100110	10100110	11110101
114	000100010100	1110010	1001011
399	001110011001	110001111	101001000
819	100000011001	1100110011	1010101010

c) Complete the following table. Use the fewest number of bits in each case: (6 pts.)

	REPRESENTATION		
Decimal	**Sign-and-magnitude**	**1's complement**	**2's complement**
-126	11111110	10000001	10000010
-103	11100111	10011000	10011001
-70	11000110	10111001	10111010
-512	11000000000	10111111111	1000000000
-39	1100111	1011000	1011001
211	011010011	011010011	011010011

PROBLEM 4 (30 PTS)

a) Perform the following additions and subtractions of the following unsigned integers. Use the fewest number of bits n to represent both operators. Indicate every carry (or borrow) from c_0 to c_n (or b_0 to b_n). For the addition, determine whether there is an overflow. For the subtraction, determine whether we need to keep borrowing from a higher bit. (8 pts)

Example (n=8):

✓ 54 + 210

	$c_8=1$	$c_7=1$	$c_6=1$	$c_5=1$	$c_4=0$	$c_3=1$	$c_2=1$	$c_1=0$	$c_0=0$	
54 = 0x36 =		0	0	1	1	0	1	1	0	+
210 = 0xD2 =		1	1	0	1	0	0	1	0	
Overflow! →	1	0	0	0	0	1	0	0	0	

✓ 77 - 194

Borrow out! →	$b_8=1$	$b_7=0$	$b_6=0$	$b_5=0$	$b_4=0$	$b_3=0$	$b_2=1$	$b_1=0$	$b_0=0$	
77 = 0x4D =		0	1	0	0	1	1	0	1	-
194 = 0xC2 =		1	1	0	0	0	0	1	0	
		0	0	0	0	1	0	1	1	

✓ 221 + 117
✓ 76 + 175

	$c_8=1$	$c_7=1$	$c_6=1$	$c_5=1$	$c_4=1$	$c_3=1$	$c_2=0$	$c_1=1$	$c_0=0$	
221 = 0xDD =		1	1	0	1	1	1	0	1	+
117 = 0x75 =		0	1	1	1	0	1	0	1	
Overflow! →	1	0	1	0	1	0	0	1	0	

No Overflow	$c_8=0$	$c_7=0$	$c_6=0$	$c_5=0$	$c_4=1$	$c_3=1$	$c_2=0$	$c_1=0$	$c_0=0$	
76 = 0x4C =		0	1	0	0	1	1	0	0	+
175 = 0xAF =		1	0	1	0	1	1	1	1	
251 = 0xFB =		1	1	1	1	1	0	1	1	

✓ 93 - 128
✓ 130 - 43

Borrow out! →	$b_8=1$	$b_7=0$	$b_6=0$	$b_5=0$	$b_4=0$	$b_3=0$	$b_2=0$	$b_1=0$	$b_0=0$	
93 = 0x5D =		0	1	0	1	1	1	0	1	-
128 = 0x80 =		1	0	0	0	0	0	0	0	
0xDD =		1	1	0	1	1	1	0	1	

No Borrow Out	$b_8=0$	$b_7=1$	$b_6=1$	$b_5=1$	$b_4=1$	$b_3=1$	$b_2=1$	$b_1=1$	$b_0=0$	
130 = 0x82 =		1	0	0	0	0	0	1	0	-
43 = 0x2B =		0	0	1	0	1	0	1	1	
87 = 0x57 =		0	1	0	1	0	1	1	1	

b) We need to perform the following operations, where numbers are represented in 2's complement: (16 pts)

✓ -93 + 128
✓ 312 + 718
✓ 87 - 62
✓ -255 - 69

- For each case:
  - ✓ Determine the minimum number of bits required to represent both summands. You might need to sign-extend one of the summands, since for proper summation, both summands must have the same number of bits.
  - ✓ Perform the binary addition in 2's complement arithmetic. The result must have the same number of bits as the summands.
  - ✓ Determine whether there is overflow by:
    - i. Using c_n, c_{n-1} (carries).
    - ii. Performing the operation in the decimal system and checking whether the result is within the allowed range for n bits, where n is the minimum number of bits for the summands.
  - ✓ If we want to avoid overflow, what is the minimum number of bits required to represent both the summands and the result?

n = 9 bits

$c_9 \oplus c_8=0$
No Overflow

	$c_9=1$	$c_8=1$	$c_7=0$	$c_6=0$	$c_5=0$	$c_4=0$	$c_3=0$	$c_2=0$	$c_1=0$	$c_0=0$
-93 =	1	1	0	1	0	0	0	1	1	+
128 =	0	1	0	0	0	0	0	0	0	
35 =	0	0	0	1	0	0	0	1	1	

-93+128 = 35 ∈ $[-2^8, 2^8-1]$ → no overflow

n = 8 bits

$c_8 \oplus c_7=0$
No Overflow

	$c_8=1$	$c_7=1$	$c_6=0$	$c_5=0$	$c_4=0$	$c_3=1$	$c_2=1$	$c_1=0$	$c_0=0$
87 =	0	1	0	1	0	1	1	1	+
-62 =	1	1	0	0	0	0	1	0	
25 =	0	0	0	1	1	0	0	1	

87-62 = 25 ∈ $[-2^7, 2^7-1]$ → no overflow

n = 11 bits

$c_{11}\oplus c_{10}=1$
Overflow!

$c_{11}=0$ $c_{10}=1$ $c_9=1$ $c_8=1$ $c_7=1$ $c_6=1$ $c_5=1$ $c_4=1$ $c_3=0$ $c_2=0$ $c_1=0$ $c_0=0$

```
312 = 0 0 1 0 0 1 1 1 0 0 0 +
718 = 0 1 0 1 1 0 0 1 1 1 0
      ---------------------
      1 0 0 0 0 0 0 0 1 1 0
```

312+718 = 1030 $\notin [-2^{10}, 2^{10}-1]$ → overflow!

To avoid overflow:
n = 12 bits (sign-extension)

$c_{12}\oplus c_{10}=0$
No Overflow

$c_{12}=0$ $c_{11}=0$ $c_{10}=1$ $c_9=1$ $c_8=1$ $c_7=1$ $c_6=1$ $c_5=1$ $c_4=1$ $c_3=0$ $c_2=0$ $c_1=0$ $c_0=0$

```
 312 = 0 0 0 1 0 0 1 1 1 0 0 0 +
 718 = 0 0 1 0 1 1 0 0 1 1 1 0
       -----------------------
1030 = 0 1 0 0 0 0 0 0 0 1 1 0
```

312+718 = 1030 $\in [-2^{11}, 2^{11}-1]$ → no overflow

n = 9 bits

$c_9\oplus c_8=1$
Overflow!

$c_9=1$ $c_8=0$ $c_7=0$ $c_6=0$ $c_5=0$ $c_4=0$ $c_3=0$ $c_2=1$ $c_1=1$ $c_0=0$

```
-255 = 1 0 0 0 0 0 0 0 1 +
 -69 = 1 1 0 1 1 1 0 1 1
       -----------------
       0 1 0 1 1 1 1 0 0
```

-255-69 = -324 $\notin [-2^8, 2^8-1]$ → overflow!

To avoid overflow:
n = 10 bits (sign-extension)

$c_{10}\oplus c_9=0$
No Overflow

$c_{10}=1$ $c_9=1$ $c_8=0$ $c_7=0$ $c_6=0$ $c_5=0$ $c_4=0$ $c_3=0$ $c_2=1$ $c_1=1$ $c_0=0$

```
-255 = 1 1 0 0 0 0 0 0 0 1 +
 -69 = 1 1 1 0 1 1 1 0 1 1
       -------------------
-324 = 1 0 1 0 1 1 1 1 0 0
```

-255-69 = -324 $\in [-2^9, 2^9-1]$ → no overflow

c) Perform the multiplication of the following numbers that are represented in 2's complement arithmetic with 4 bits. (6 pts)

✓ 0101×0101, 1000×0111, 0111×1001.

```
        0 1 0 1 x
        0 1 0 1
        -------
        0 1 0 1
      0 0 0 1
    0 1 0 1
  0 0 0 0
  ---------------
  0 0 0 1 1 0 0 1
```

```
1 0 0 0 x  ➡        1 0 0 0 x
0 1 1 1             0 1 1 1
                    -------
                    1 0 0 0
                  1 0 0 0
                1 0 0 0
              0 0 0 0
              ---------------
              0 0 1 1 1 0 0 0
                     ⬇
              1 1 0 0 1 0 0 0
```

```
0 1 1 1 x  ➡        0 1 1 1 x
1 0 0 1             0 1 1 1
                    -------
                    0 1 1 1
                  0 1 1 1
                0 1 1 1
              0 0 0 0
              ---------------
              0 0 1 1 0 0 0 1
                     ⬇
              1 1 0 0 1 1 1 1
```

PROBLEM 5 (10 PTS)

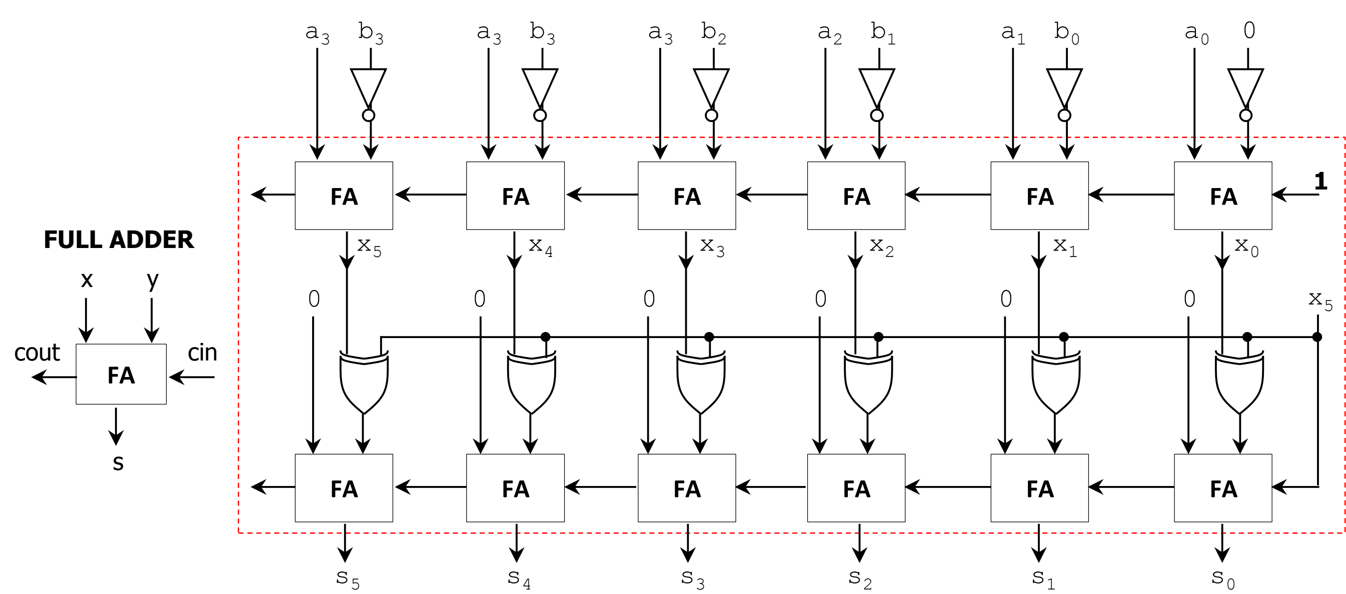
- Given two 4-bit signed (2's complement) numbers A, B, sketch the circuit that computes $|A-2\times B|$. You can only use full adders and logic gates. Make sure your circuit works for all cases. If there is overflow, design your circuit so that the final answer is always the correct one with the correct number of bits (i.e., overflow must be avoided).

$A = a_3a_2a_1a_0$, $B = b_3b_2b_1b_0$
$A, B \in [-8,7] \rightarrow 2B \in [-16,14]$ requires 5 bits in 2C.

✓ $X = A - 2B \in [-22,23]$ requires 6 bits in 2C. Thus, the operation $A-2B$ requires 6 bits (we sign-extend A and $2B$).

✓ $|X| = |A-2B| \in [0,23]$ requires 6 bits in 2C. Thus, the second operation $0 \pm X$) only requires 6 bits.

- If $x_5 = 1 \rightarrow X < 0 \rightarrow$ we do $0 - X$.
- If $x_5 = 0 \rightarrow X \geq 0 \rightarrow$ we do $0 + X$.



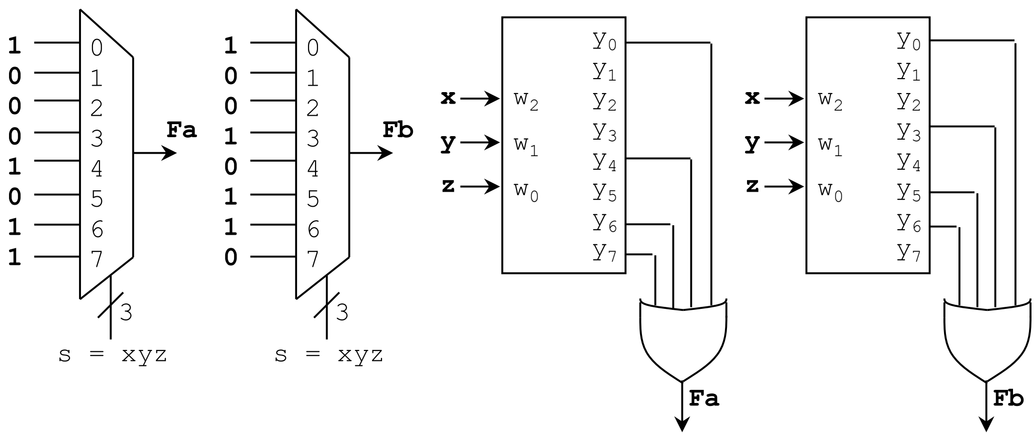
PROBLEM 6 (20 PTS)

a) Implement the following functions using i) decoders (and OR gates) and ii) multiplexers: (5 pts)

- ✓ $F_a = (\overline{Y+Z}) \oplus (XY)$
- ✓ $F_b = (\overline{\bar{X} \oplus Y}) \oplus \bar{Z}$

w_2 w_1 w_0

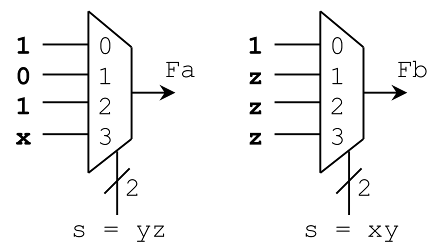
x	y	z	Fa	Fb
0	0	0	1	1
0	0	1	0	0
0	1	0	0	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	1	1
1	1	1	1	0



b) Using only a 4-to-1 MUX (do not use NOT gates), implement the following functions. (5 pts)

- ✓ $F_a(X,Y,Z) = \sum(m_0, m_2, m_4, m_6, m_7)$
- ✓ $F_b(X,Y,Z) = \prod(M_2, M_4, M_6)$

x	y	z	Fa	Fb
0	0	0	1	1
0	0	1	0	1
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	1	0
1	1	1	1	1



c) Complete the timing diagram of the circuit shown below. Note that $x = x_1x_0$, $y = y_3y_2y_1y_0$ (10 pts)