Solutions - Midterm Exam

(October 18th @ 5:30 pm)

Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (24 PTS)

a) Complete the following table. The decimal numbers are unsigned: (5 pts.)

Decimal	BCD	Binary	Reflective Gray Code
37	00110111	100101	110111
126	000100100110	1111110	1000001
69	01101001	1000101	1100111

b) Complete the following table. The decimal numbers are signed. Use the fewest number of bits in each case: (15 pts.)

	REPRESENTATION		
Decimal	**Sign-and-magnitude**	**1's complement**	**2's complement**
-63	1111111	1000000	1000001
-16	110000	101111	10000
-11	11011	10100	10101
22	010110	010110	010110
-32	1100000	1011111	100000
0	00	11111	0

c) Convert the following decimal numbers to their 2's complement representations. (4 pts)

✓ 17.125
+17.125 = 010001.001

✓ -16.625
+16.625 = 010000.101
⇒ -16.625 = 101111.011

PROBLEM 2 (12 PTS)

- The figure below depicts the entire memory space of a microprocessor. Each memory address occupies one byte. 1KB = 2^{10} bytes, 1MB = 2^{20} bytes, 1GB = 2^{30} bytes

✓ What is the size (in bytes, KB, or MB) of the memory space? What is the address bus size of the microprocessor? (3 pts.)

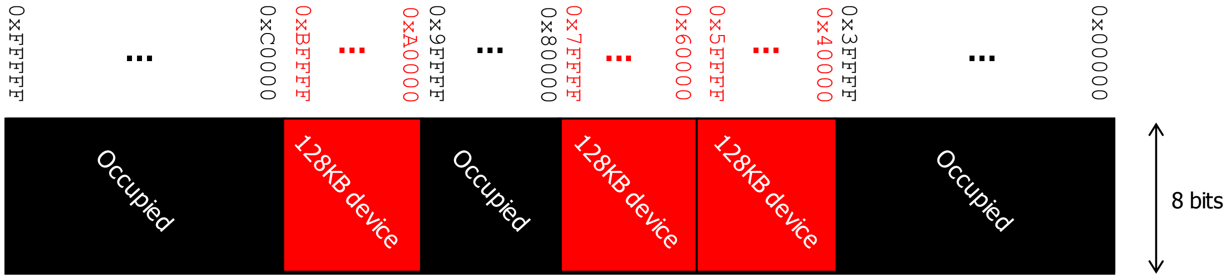
Address space: 0x00000 to 0xFFFFF. To represent all these addresses, we require 20 bits. So, the address bus size of the microprocessor is 20 bits. The size of the memory space is then 2^{20} = 1 MB.

✓ If we have a memory chip of 128 KB, how many bits do we require to address those 128 KB of memory? (1 pt.)

128 KB memory device: 128KB = 2^{17} bytes. Thus, we require 17 bits to address the memory device.

✓ We want to connect the 128 KB memory chip to the microprocessor. For optimal implementation, we must place those 128 KB in an address range where every address shares some MSBs. Provide a list of all the possible address ranges that the 128 KB memory chip can occupy. You can only use the non-occupied portions of the memory space as shown below.

▫ 0x40000 to 0x5FFFF ▫ 0x60000 to 0x7FFFF ▫ 0xA0000 to 0xBFFFF



PROBLEM 3 (20 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. (6 pts)

✓ 49 + 17

✓ 22 - 33

```
                  c6=1 c5=1 c4=0 c3=0 c2=0 c1=1 c0=0
49 = 0x31 =            1    1    0    0    0    1  +
17 = 0x10 =            0    1    0    0    0    1
Overflow! →       1    0    0    0    0    1    0
```

```
Borrow out! →  b6=1 b5=0 b4=0 b3=0 b2=0 b1=1 b0=0
22 = 0x16 =         0    1    0    1    1    0  -
33 = 0x21 =         1    0    0    0    0    1
                    1    1    0    1    0    1
```

b) Perform the following operations, where numbers are represented in 2's complement. Indicate every carry from c_0 to c_n. For each case, use the fewest number of bits to represent the summands and the result so that overflow is avoided. (11 pts)

- ✓ -61 − 18

```
n = 7 bits       c7=1 c6=0 c5=0 c4=1 c3=1 c2=1 c1=0 c0=0
c7⊕c6=1   -61 =       1    0    0    0    0    1    1  +
Overflow! -18 =       1    1    0    1    1    1    0
                      0    1    1    0    0    0    1
```

$-61 -18 = -79 \notin [-2^6, 2^6-1]$ → overflow!

To avoid overflow: n = 8 bits (sign-extension)

```
c8⊕c7=0      c8=1 c7=1 c6=0 c5=0 c4=1 c3=1 c2=1 c1=0 c0=0
No Overflow
-61 =             1    1    0    0    0    0    1    1  +
-18 =             1    1    1    0    1    1    1    0
                  1    0    1    1    0    0    0    1
```

$-61 -18 = -79 \in [-2^7, 2^7-1]$ → no overflow

- ✓ 22 − 33

```
n = 7 bits       c7=0 c6=0 c5=1 c4=1 c3=1 c2=1 c1=0 c0=0
c7⊕c6=0     22 =      0    0    1    0    1    1    0  +
No Overflow 33 =      1    0    1    1    1    1    1
                      1    1    1    0    1    0    1
```

$22 - 33 = -11 \in [-2^6, 2^6-1]$ → no overflow

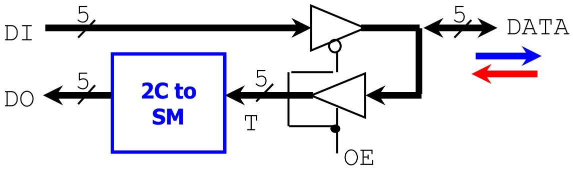
c) Perform binary multiplication of the following numbers that are represented in 2's complement arithmetic with 4 bits. (3 pts)

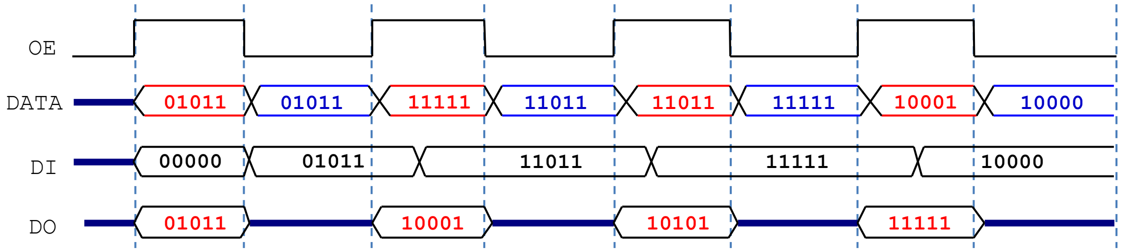
- ✓ -7 x 6.

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

PROBLEM 4 (12 PTS)

- Complete the timing diagram (signals *DO* and *DATA*) of the following circuit. The circuit in the blue box treats the input T as a 5-bit signed (2C) number and converts it to the sign-and-magnitude representation with 5 bits.
  - ✓ Example: if T = 10110, then DO = 11010.

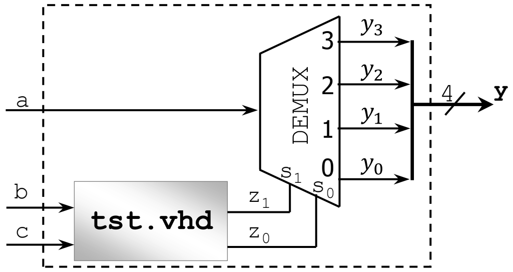




PROBLEM 5 (15 PTS)

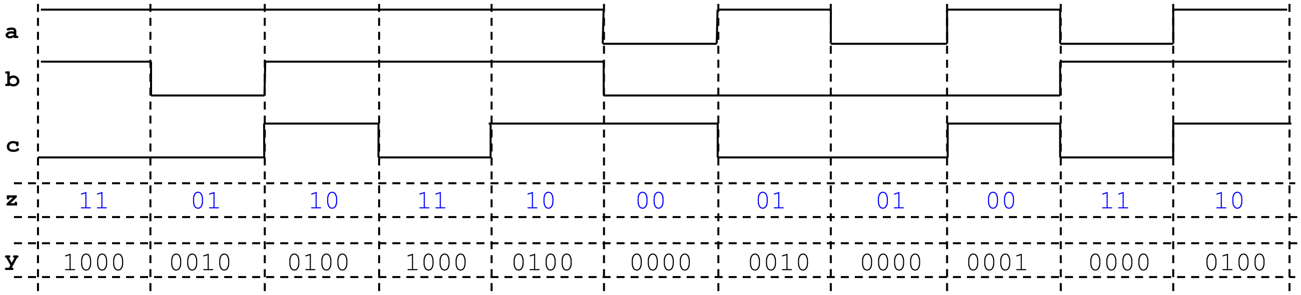
- Complete the timing diagram of the following circuit. The VHDL code (`tst.vhd`) corresponds to the shaded circuit.

$z = z_1z_0$, $y = y_3y_2y_1y_0$



```
library ieee;
use ieee.std_logic_1164.all;

entity tst is
  port (b,c : in std_logic;
        z: out std_logic_vector(1 downto 0));
end tst;
```

```
architecture bhv of tst is

begin

  process (b, c)
  begin
     z <= b & '0';
     if c = '0' then
        z <= b & (not(c));
     end if;
  end process;

end bhv;
```



z	11	01	10	11	10	00	01	01	00	11	10
y	1000	0010	0100	1000	0100	0000	0010	0000	0001	0000	0100

- Get the Boolean equations for y_3, y_2, y_1, y_0 based on a, b, c (4 pts)

$z_1 = b$, $z_0 = \bar{c}$

$y_3(a,b,c) = z_1z_0.a = b\bar{c}a$

$y_2(a,b,c) = z_1\bar{z_0}.a = bca$

$y_1(a,b,c) = \bar{z_1}z_0.a = \bar{b}\bar{c}a$

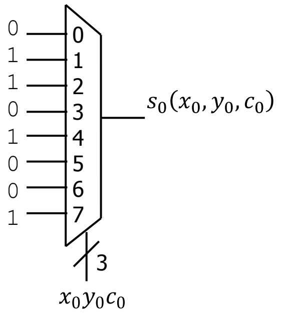
$y_0(a,b,c) = \bar{z_1}\bar{z_0}.a = \bar{b}ca$

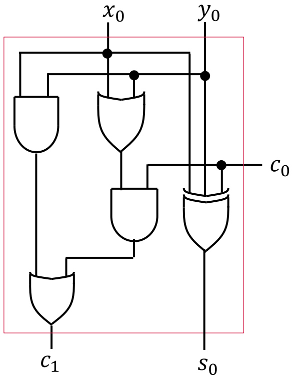
PROBLEM 6 (17 PTS)

- Given the circuit in the figure:
  - ✓ Implement s_0 using ONLY an 8-to-1 MUX. (5 pts.)

$s_0(x_0, y_0, c_0) = x_0 \oplus y_0 \oplus c_0$

x_0	y_0	c_0	c_1	s_0
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	1	0
1	1	0	1	0
1	1	1	1	1





  - ✓ Implement c_1 using ONLY 2-to-1 MUXs (AND, OR, NOT, XOR gates are not allowed). (12 pts)

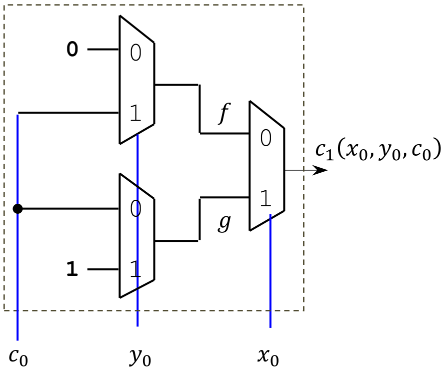
$c_1(x_0, y_0, c_0) = x_0y_0 + x_0c_0 + y_0c_0$

$c_1(x_0, y_0, c_0) = \overline{x_0}c_1(0, y_0, c_0) + x_0c_1(1, y_0, c_0) = \overline{x_0}(y_0c_0) + x_0(y_0 + c_0 + y_0c_0)$

$c_1(x_0, y_0, c_0) = \overline{x_0}g(y_0, c_0) + x_0h(y_0, c_0)$

$g(y_0, c_0) = \overline{y_0}g(0, c_0) + y_0g(1, c_0) = \overline{y_0}(0) + y_0(c_0)$

$h(y_0, c_0) = \overline{y_0}h(0, c_0) + y_0h(1, c_0) = \overline{y_0}(c_0) + y_0(1)$


0
0
1
1
0
1
f
g
0
1
$c_1(x_0, y_0, c_0)$
c_0
y_0
x_0