Solutions - Homework 2

(Due date: February 1st @ 5:30 pm)

Presentation and clarity are very important! Show your procedure!

PROBLEM 1 (32 PTS)

- In ALL these problems (a, b, c), 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. (12 pts)
  ✓ -31.3125, 37.5078125, -256.65625, -391.25.
    - +31.3125 = 011111.0101 → -31.3125 = 100000.1011 = 0xE0.B
    - +37.5078125 = 0100101.1000001 = 0x25.82
    - +256.65625 = 0100000000.10101 → -256.65625 = 1011111111.01011 = 0xEFF.58
    - +391.25 = 0110000111.01 → -391.25 = 1001111000.11 = 0xE78.C

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

Decimal	BCD	Binary	Reflective Gray Code
397	001110010111	110001101	101001011
318	001100011000	100111110	110100001
835	100000110101	1101000011	1011100010
256	001001010110	100000000	110000000
232	001000110010	11101000	10011100
114	000100010100	1110010	1001011
206	001000000110	11001110	10101001
259	001001011001	100000011	110000010

  c) Complete the following table. Use the fewest number of bits in each case: (12 pts.)

	REPRESENTATION		
Decimal	Sign-and-magnitude	1's complement	2's complement
-133	110000101	101111010	101111011
-256	1100000000	1011111111	100000000
-152	110011000	101100111	101101000
-85	11010101	10101010	10101011
-52	1110100	1001011	1001100
105	01101001	01101001	01101001

PROBLEM 2 (20 PTS)

a) What is the minimum number of bits required to represent: (2 pts)
  ✓ Memory addresses from 0 to 8192?
  $\lceil \log_2(8192+1) \rceil = 14$
  ✓ 32767 symbols?
  ✓ $\lceil \log_2(32767) \rceil = 15$

b) A microprocessor has a 28-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. (6 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

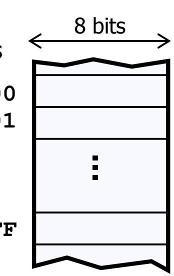
    Address Range: 0x0000000 to 0xFFFFFFF.
    With 28 bits, we can address 2^{28} bytes, thus we have $2^{20}2^{8}$ = 256 MB of address space.

  - A memory device is connected to the microprocessor. Based on the size of the memory, the microprocessor has assigned the addresses 0xB1C0000 to 0xB1FFFFF 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 address in the given range (where the memory is located).

    Thus, the size of the memory is 2^{18} = 256 KB.

```
1011 0001 1100 0000 0000 0000 0000: 0xB1C00000
1011 0001 1100 0000 0000 0000 0001: 0xB1C00001
   ...
   ...
   ...
1101 0001 1111 1111 1111 1111 1111: 0xB1CFFFFF
```



c) The figure below depicts the entire memory space of a microprocessor. Each memory address occupies one byte. (12 pts)
- What is the size (in bytes, KB, or MB) of the memory space? What is the address bus size of the microprocessor?

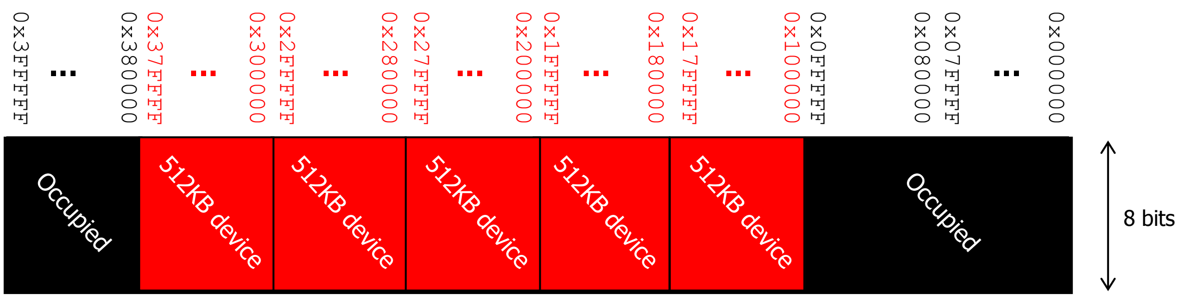
  Address space: `0x000000` to `0x3FFFFF`. To represent all these addresses, we require 22 bits. So, the address bus size of the microprocessor is 22 bits. The size of the memory space is then 2^{22} = 4 MB.

- If we have a memory chip of 512KB, how many bits do we require to address 512KB of memory?

  512KB = $2^9 2^{10}$ bytes. Thus, we require 19 bits to address only this memory device.

- We want to connect the 512KB memory chip to the microprocessor. Provide a list of all the possible address ranges that the 512KB memory chip can occupy. You can only use the non-occupied portions of the memory space as shown below.

```
0x100000 to 0x17FFFF
0x180000 to 0x1FFFFF
0x200000 to 0x27FFFF
0x280000 to 0x2FFFFF
0x300000 to 0x37FFFF
```



PROBLEM 3 (38 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 byte. (8 pts)

Example (n=8):

✓ 54 + 210

```
                 c8=1 c7=1 c6=1 c5=1 c4=0 c3=1 c2=1 c1=0 c0=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! -> b8=1 b7=0 b6=0 b5=0 b4=0 b3=0 b2=1 b1=0 b0=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
```

✓ 211 + 99
✓ 101 + 35

n = 8 bits

```
Overflow!        c8=1 c7=1 c6=0 c5=0 c4=0 c3=0 c2=1 c1=1 c0=0
211 = 0xD3 =           1    1    0    1    0    0    1    1   +
 99 = 0x63 =           0    1    1    0    0    0    1    1
310 =0x136 =      1    0    0    1    1    0    1    1    0
```

n = 7 bits

```
Overflow!        c7=1 c6=1 c5=0 c4=0 c3=1 c2=1 c1=1 c0=0
101 = 0x65 =           1    1    0    0    1    0    1   +
 35 = 0x23 =           0    1    0    0    0    1    1
136 = 0x88 =      1    0    0    0    1    0    0    0
```

✓ 51 - 96
✓ 256 - 57

n = 7 bits

```
Borrow out! -> b7=1 b6=0 b5=0 b4=0 b3=0 b2=0 b1=0 b0=0
51 = 0x33 =          0    1    1    0    0    1    1   -
96 = 0x60 =          1    1    0    0    0    0    0
     0xD3 =          1    0    1    0    0    1    1
```

n = 9 bits

```
No Borrow Out    b9=0 b8=1 b7=1 b6=1 b5=1 b4=1 b3=1 b2=1 b1=1 b0=0
256 = 0x100 =          1    0    0    0    0    0    0    0    0   -
 57 = 0x039 =          0    0    0    1    1    1    0    0    1
199 = 0x0C7 =          0    1    1    0    0    0    1    1    1
```

b) We need to perform the following operations, where numbers are represented in 2's complement: (24 pts)

- ✓ `-77 + 216`
- ✓ `-129 + 128`
- ✓ `313 + 711`
- ✓ `-62 + 99`
- ✓ `-122 - 26`
- ✓ `167 + 512`

- 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

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

```
-77 = 1 1 0 1 1 0 0 1 1 +
216 = 0 1 1 0 1 1 0 0 0
-------------------------
139 = 0 1 0 0 0 1 0 1 1
```

$-77+216 = 139 \in [-2^8, 2^8-1] \rightarrow$ no overflow

n = 9 bits

$c_9 \oplus c_8 = 0$ No Overflow

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

```
-129 = 1 0 1 1 1 1 1 1 1 +
 128 = 0 1 0 0 0 0 0 0 0
--------------------------
  -1 = 1 1 1 1 1 1 1 1 1
```

$-129+128 = -1 \in [-2^8, 2^8-1] \rightarrow$ no overflow

n = 11 bits

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

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

```
313 =  0 0 1 0 0 1 1 1 0 0 1 +
711 =  0 1 0 1 1 0 0 0 1 1 1
------------------------------
       1 0 0 0 0 0 0 0 0 0 0
```

$313+711 = 1024 \notin [-2^{10}, 2^{10}-1] \rightarrow$ overflow!

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

$c_{12} \oplus c_{11} = 0$ No Overflow

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

```
 313 = 0 0 0 1 0 0 1 1 1 0 0 1 +
 711 = 0 0 1 0 1 1 0 0 0 1 1 1
---------------------------------
1024 = 0 1 0 0 0 0 0 0 0 0 0 0
```

$313+711 = 1024 \in [-2^{11}, 2^{11}-1] \rightarrow$ no overflow

n = 8 bits

$c_8 \oplus c_7 = 1$ Overflow!

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

```
-122 = 1 1 0 0 0 0 1 0 +
 -26 = 1 1 1 0 0 1 1 0
------------------------
       0 1 1 0 1 1 0 0
```

$-122-26 = -148 \notin [-2^7, 2^7-1] \rightarrow$ overflow!

To avoid overflow:
n = 9 bits (sign-extension)

$c_9 \oplus c_8 = 0$ No Overflow

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

```
-122 = 1 1 1 0 0 0 0 1 0 +
 -26 = 1 1 1 1 0 0 1 1 0
--------------------------
       1 0 1 1 0 1 1 0 0
```

$-122-26 = -148 \in [-2^8, 2^8-1] \rightarrow$ no overflow

n = 8 bits

$c_8 \oplus c_7 = 0$ No Overflow

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

```
-62 =  1 1 0 0 0 0 1 0 +
 99 =  0 1 1 0 0 0 1 1
------------------------
 37 =  0 0 1 0 0 1 0 1
```

$-62+99 = 37 \in [-2^7, 2^7-1] \rightarrow$ no overflow

n = 11 bits

$c_{11} \oplus c_{10} = 0$ No Overflow

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

```
167 =  0 0 0 1 0 1 0 0 1 1 1 +
512 =  0 1 0 0 0 0 0 0 0 0 0
------------------------------
679 =  0 1 0 1 0 1 0 0 1 1 1
```

$167+512 = 679 \in [-2^{10}, 2^{10}-1] \rightarrow$ no overflow

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

- ✓ `0101×0100, 1001×0111, 1011×1101.`

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

```
1 0 1 1 x   ➡        0 1 0 1 x
1 1 0 1                0 0 1 1
                     ---------
                       0 1 0 1
                     0 1 0 1
                   0 0 0 0
             -----------------
             0 0 1 1 1 1
                  ↓
             0 0 1 1 1 1
```

PROBLEM 4 (10 PTS)

- Complete the timing diagram (signals DO and $DATA$) of the following circuit. The circuit in the blue box computes the unsigned summation `T+6`, with the result having 5 bits.
  For example: if `T=1001` → `DO=1001 + 0110 = 01111. If T=1100` → `DO=1101+0110 = 10011.`