EECE.3170: Microprocessor Systems Design I

Summer 2016

Homework 1 Solution

1. *(50 points) Given each of the binary or hexadecimal number below, determine what the decimal value is if the number is (i) an unsigned integer, and (ii) a signed integer. Note that, in some cases, your answers for both will be the same.*

a. *01101001_2*

Since MSB = 0, value is same whether unsigned or signed—figure out the significance of each position in which a bit = 1, and sum those values together.

$01101001_2 = 64 + 32 + 8 + 1 = \mathbf{105}$

b. *10100110_2*

For an unsigned integer, we use the same method as in part (a)

$10100110_2 = 128 + 32 + 4 + 2 = \mathbf{166}$

For a signed integer, recognize that this value is negative; to find its magnitude, take the two's complement:

$-10100110_2 = 01011001_2 + 1 = 01011010_2 = 64 + 16 + 8 + 2 = 90$

Therefore, 10100110_2 = **-90** when treated as a signed integer.

c. *8Eh (or 0x8E—recall that, in x86 assembly notation, the "h" at the end of a number signifies that the previous value is in hexadecimal)*

For an unsigned integer, we don't really need to convert to binary; if you want to do so, 8Eh = 10001110_2. However, we can also just convert directly to decimal:

$8E_{16} = (8 \text{ x } 16) + (14 \text{ x } 1) = \mathbf{142}$

As a signed integer, note that this value is negative, since its MSB = 1. To find the magnitude, once again take the two's complement:

$-10001110_2 = 01110001_2 + 1 = 01110010_2 = 64 + 32 + 16 + 2 = 114$

Therefore, as a signed integer, 8Eh = **-114.**

d. 6AD7h

Since the most significant bit of this number is 0 (6AD7h = $0110\ 1010\ 1101\ 0111_2$), it has the same value whether it is treated as a signed or unsigned integer. That value is:

$(6 \text{ x } 16^3) + (10 \text{ x } 16^2) + (13 \text{ x } 16^1) + (7 \text{ x } 16^0) =$
$(6 \text{ x } 4096) + (10 \text{ x } 256) + (13 \text{ x } 16) + (7 \text{ x } 1) = 24576 + 2560 + 208 + 7 =$ **27351**

e. CAB5h

This number has different values when treated as signed or unsigned, since the MSB is 1 (CAB5h = $1100\ 1010\ 1011\ 0101_2$). As an unsigned integer:

$(12 \text{ x } 16^3) + (10 \text{ x } 16^2) + (11 \text{ x } 16^1) + (5 \text{ x } 16^0) =$
$(12 \text{ x } 4096) + (10 \text{ x } 256) + (11 \text{ x } 16) + (5 \text{ x } 1) =$
$49152 + 2560 + 176 + 5 =$ **51893**

As a signed integer, the magnitude is:

-CAB5h = $-1100\ 1010\ 1011\ 0101_2 = 0011\ 0101\ 0100\ 1011_2$ = 354Bh

I've shown the conversion back into hexadecimal because it might be slightly easier to figure out the decimal value of the magnitude using what we already know about converting a 16-bit value from hex to decimal:

$(3 \text{ x } 16^3) + (5 \text{ x } 16^2) + (4 \text{ x } 16^1) + (11 \text{ x } 16^0) =$
$(3 \text{ x } 4096) + (5 \text{ x } 256) + (4 \text{ x } 16) + (11 \text{ x } 1) = 12288 + 1280 + 64 + 11 = 13643$

Therefore, CAB5h = **-13643** as a signed integer.

2. *(50 points) Assume the contents of memory are shown below. All values are in hexadecimal. The table shows four bytes per line; the given address is the starting address of each line.*

   Each block in the table contains a single byte, with the low and high bytes per line indicated as shown. Each byte has its own address, so the byte at address 11570h is 20h, address 11571h is 16h, address 11572h is EEh, and address 11573h is CEh.

   You should assume all multi-byte values are stored in little-endian format.

Address	Lo			Hi
11570h	20	16	EE	**CE**
11574h	**31**	70	**FF**	**EF**
11578h	01	4E	**DB**	**AB**
1157Ch	**CF**	**09**	49	22
11580h	55	**15**	**3A**	**68**
11584h	**3B**	**87**	29	D7
11588h	51	30	B2	95

For each address and amount of data listed, answer the following:

- *What data are stored at that address?*
- *Would an access to the given amount of data at that address be aligned?*
- *If the data represents a signed integer, what is the sign of that value?*

For example, given "Address: 11570h, Data size: word," your response would be that the word at 11570h is 1620h, the access is aligned, and the data represents a positive integer.

<u>**Note:**</u> The key points to remember for this problem are:

- Little-endian data are stored with the least significant byte at the lowest address.
- An access is aligned if the address is divisible by the number of bytes being accessed.
- In signed formats, the integer is positive if the most significant bit is 0 and negative if that bit is 1.

a. *Address: 11576h, Data size: word*

The word at this address is **EFFFh**, the access <u>is aligned</u> (since 11576h is divisible by 2), and the data represents a <u>negative</u> integer, since its most significant bit is 1.

b. *Address: 11581h, Data size: byte*

The byte at this address is **15h**, the access <u>is aligned</u> (since every address is divisible by 1), and the data represents a <u>positive</u> integer, since its most significant bit is 0.

c. *Address: 1157Ah, Data size: double word*

The double word at this address is **09CFABDBh**, the access <u>is not aligned</u> (since 1157Ah is not divisible by 4), and the data represents a <u>positive</u> integer, since its most significant bit is 0.

d. Address: 11573h, Data size: word

The word at this address is **31CEh**, the access is not aligned (since 11573h is not divisible by 2), and the data represents a positive integer, since its most significant bit is 0.

e. Address: 11582h, Data size: double word

The double word at this address is **873B683Ah**, the access is not aligned (since 11582h is not divisible by 4), and the data represents a negative integer, since its most significant bit is 1.