Homework 1 - Jeremy Tandjung

Q1

1) $9730\,4_{16}$ → base 17

① $9730\,4_{16}$ → base 10

$$4 \cdot 16^0 + \underset{\text{D}}{13} \cdot 16^1 + 3 \cdot 16^2 + 7 \cdot 16^3 + 9 \cdot 16^4$$

$$= 4 + 208 + 768 + 28\,672 + 589{,}824$$

589 824
28 672 +
618 496
768 +
619, 264
208 +
619 472
+1
619 476$_{10}$

② 619476_{10} → base 17

$619476 \;/\; 17 = 36439$

% 17 = 13 -> (D)

36439 / 17 = 2143

% 17 = (8)

2143 / 17 = 126

% 17 = (1)

126 / 17 = 7

% 17 = (7)

7 / 17 = 0

% 17 = (7)

77189_{16}

2) $C56E_{17}$ -> dec

A -> 10
B -> 11
C -> 12
D -> 13
E -> 14
F -> 15
G -> 16

$12 \cdot 17^3 + 5 \cdot 17^2 + 16 \cdot 17^1 + 14 \cdot 17^0 =$

$12 \cdot 17^3 =$ 58956

$5 \cdot 17^2 =$ 1445 +

60401

$16 \cdot 17^1 =$ 272 +

60673

$14 \cdot 17^0 =$ 14 +

60687_{10}

Q2 8-bit system

1) -102

102 -> 01100110

flip -> 10011001

1 +

10011010

$-102_{10} =$ $9A_{16}$

-87

87 -> 01010111

flip -> 10101000

+ 1

10101001

$-87_{10} = A9_{16}$

2) a) $9A + A9 = 143_{16}$

8-bit

$143_{16} = 0001\ 0100\ 0011_2$

overflow

$= 323_{16}$

b) It's wrong

because we're using an 8-bit system, our range of possible numbers is from -128_{10} to 127_{10}

When calculated, the actual answer is 193_{16} which has 9-bit places (out of bounds), as expected when calculated using binary, it will give an overflow

From looking at the decimal answer (323_{10}) we can also determin it's an overflow.

Q3

.1) $-69/32 =$

-2.15625

① Convert to bin

$2_{10} \rightarrow 10_2$

$.15625 * 2 = 0.3125 + 0$

$.3125 * 2 = 0.625 + 0$

$0.625 * 2 = .25 + 1$

$.25 * 2 = .5 + 0$

$.5 * 2 = 0 + 1$

-10.00101

② normalize

$-1.00010 1 \times 2^1$

③ Convert

Sign $\rightarrow$ 1

exponent $\rightarrow$ $127 + 1 = 128$

$= 1000\ 0000$

Mantisa $\rightarrow$ 000 101 0000 0000 0000 00000

∴

1100 0000 0000 1010 0000 0000 0000 :

0x C 0 0 A 0 0 0 0

-69/32 = 0x C00A0000

b) 13.625

↓

① 1101

.625 * 2 = .25 + 1

.25 * 2 = .5 + 0

.5 * 2 = 0 + 1

1101 . 101

② normalize

$1.101101 \cdot 2^3$

③ Convert

sign -> 0

exponent -> 127 + 3 = 130

1000 0010

Matrissa -> 101101 0000000000000000

0100 0001 0101 1010 0000 0000 0000 0000

0x 41 5A 0000

2) 42E48000

0100 0010 1110 0100 1000 0000 0000 0000

(+)

↓

exponent = 128 + 4 + 1 − 127

= 6

Matrissa = 110 0100 1000 0000 0000 0000

decimal point → .11001001

+ 1.11001001 · 2^6

= 1110010.01

64 + 32 + 16 + 2 = 114 1 × 0.25 = 0.25

∴ 14.25_{10}

b) C6F00040

1100 0110 1111 0000 0000 0000 0100 0000

Sign -> 1 -> (−)

exponent -> 1000 0110

$128 + 8 + 4 + 1 - 127 = 14$

Matrissa -> 1110 0000 0000 0000 0100 0000

$-1.1110000000000001 \times 2^{14}$

$= -11110000000000.001$

Integer -> $7.16^3 + 8.16^2 =$

$7.16^3 = 28672$

$8.16^2 = 2048 +$

30720

decimal place -> $0*2^{-1} + 0*2^{-2} + 1*2^{-3}$

$= 0.125$

$\therefore -30720.125_{10}$ //

Q4

The word data in $4000 is 45 15.

What I did was,

1. Rewrite source code
2. Run simulation
3. Step over each step, till before the loop starts over
4. Open memory view
5. Go to address $4000
6. Look up 2 bytes from $4000 (because it's a word)