Homework Assignment 2

CS/ECE 3810: Computer Organization
Sept 10, 2020

Kyle Kazemini

Due Date: Sept 16, 2020.
(100 points)

Important Notes:

- Solutions turned in must be your own. Please, mention references (if any) at the end of each question. *Please refrain from cheating.*
- All solutions must be accompanied by the equations used/logic/intermediate steps. Writing only the final answer will receive **zero** credits.
- Partial score of every question is dedicated to each correct final answer provided by you. Please ensure both your equation/logic and final answer are correct. Moreover, you are expected to provide explanation for your solutions.
- All units must be mentioned wherever required.
- Late submissions **(after 11:59PM on 09/16/2020)** will not be accepted.
- We encourage all solutions to be typed in for which you could use software programs like LaTeX, Microsoft Word etc. If you submit handwritten solutions, they must be readable by the TAs to receive credits.
- All submitted solutions must be in the PDF format unless otherwise mentioned.

Question 1. In this problem, we will practice representing numbers in different number systems–i.e., decimal, binary, hexadecimal, and octal. Assume that all the numbers mentioned below are unsigned. [Refer to MIPS ISA I Lecture Slides 15 - 21]

i) Convert the binary numbers 10110110_{bin} and 1001011_{bin} to decimal and hexadecimal formats. Show your work in detail. **(6 points)**

$$10110110_{bin} = 0*2^0 + 1*2^1 + 1*2^2 + 0*2^3 + 1*2^4 + 1*2^5 + 0*2^6 + 1*2^7 =$$
$$= 0 + 2 + 4 + 0 + 16 + 32 + 0 + 128 = 182_{dec}$$

$10110110_{bin} = \underbrace{1011}_{11}\ \underbrace{0110}_{6}$ 11 in hex is B, 6 in hex is 6, so $10110110_{bin} = B6_{hex}$

$1001011_{bin} = 1*2^0 + 1*2^1 + 0*2^2 + 1*2^3 + 0*2^4 + 0*2^5 + 1*2^6 =$
$= 1+2+0+8+0+0+64 = 75_{dec}$

$1001011_{bin} = \underbrace{0100}_{4}\ \underbrace{1011}_{11}$ 4 in hex is 4, 11 in hex is B, so $1001011_{bin} = 4B_{hex}$

ii) Convert the octal numbers 51_{oct} and 44_{oct} to decimal and hexadecimal formats. Show your work in detail. **(6 points)**

$51_{oct} = 5*8^1 + 1*8^0 = 40+1 = 41_{dec}$

$51_{oct} = \underbrace{101}_{5}\ \underbrace{001}_{1} \Rightarrow 0010\ 1001 = 29_{hex}$

$44_{oct} = 4*8^1 + 4*8^0 = 32+4 = 36_{dec}$

$44_{oct} = \underbrace{100}_{4}\ \underbrace{100}_{4} \Rightarrow 0010\ 0100 = 24_{hex}$

iii) Convert the hexadecimal numbers $FEED_{hex}$ and $BEEF_{hex}$ to decimal and octal formats. Show your work in detail. **(6 points)**

$FEED_{hex} = D*16^0+E*16^1+E*16^2+F*16^3 = 13+14*16+14*256+15*4096 = 65,261_{dec}$

$FEED_{hex} = \underbrace{1111}_{F}\ \underbrace{1110}_{E}\ \underbrace{1110}_{E}\ \underbrace{1101}_{D} \Rightarrow 001\ 111\ 111\ 011\ 101\ 101 = 177,355_{oct}$

$BEEF_{hex} = F*16^0+E*16^1+E*16^2+B*16^3 = 15+14*16+14*256+11*4096 = 48879$

$BEEF_{hex} = \underbrace{1011}_{B}\ \underbrace{1110}_{E}\ \underbrace{1110}_{E}\ \underbrace{1111}_{F} \Rightarrow 001\ 011\ 111\ 011\ 101\ 111 = 137,357_{oct}$

iv) Convert the decimal numbers 3810_{dec} and 2020_{dec} to binary and hexadecimal. Show your work in detail. **(6 points)**

$3810_{dec} = 3810/16 = 238/16 = 14/16 = 0$
Using the remainders from those integer divisions: $3810_{dec} = EE2_{hex}$

$3810_{dec} = 3810/2 = 1905/2 = 952/2 = 476/2 = 238/2 = 119/2 = 59/2 = 29/2 = 14/2 = 7/2 = 3/2 = 1/2 = 0$
Using the remainders from those integer divisions: $3810_{dec} = 111011100010_{bin}$

$2020_{dec} = 2020/16 = 126/16 = 7/16 = 0$ Using the remainders from those integer divisions: $2020_{dec} = 7E4_{hex}$

Representing student UID in different formats. Each student has a unique UID number at University of Utah, which is a combination of the letter 'u' followed by a 7-digit number. Convert the least significant 5 digits of this number, which is in the decimal system into binary, hexadecimal, and octal number systems [Refer to MIPS ISA I Lecture Slides 15 - 21].

For example, in u1234567, 34567 are the 5 least significant digits. Below are the binary, hexadecimal, and octal representations of 34567_{dec}.

Binary representation: 1000011100000111_{bin}
Hexadecimal representation: 8707_{hex}
Octal representation: 103407_{oct}

Question 2. In this problem, we will practice representing the 5 least significant digits of **your UID** in the binary, hexadecimal, and octal number systems.

Fill the table below with appropriate representations. You need to show your work. [Refer to MIPS ISA I Lecture Slides 15 - 21]. **(20 points)**

My uid: u1127157, so the 5 least significant digits are 27157_{dec}

$27157_{dec} = 13578/2 = 6789/2 = 3394/2 = 1679/2 = 848/2 = 424/2 = 212/2 = 106/2 = 53/2 = 26/2 = 13/2 = 6/2 = 3/2 = 1/2 = 0$
Using the remainders from those integer divisions, $27157_{dec} = 110101000010101_{bin}$

$27157_{dec} = 3394/8 = 424/8 = 53/8 = 6/8 = 0$
Using the remainders from those integer divisions, $27157_{dec} = 65025_{oct}$

$27157_{dec} = 1697/16 = 106/16 = 6/16 = 0$
Using the remainders from those integer divisions, $27157_{dec} = 6A15_{hex}$

UID	The 5 least significant digits (Decimal)	Binary	Hexadecimal	Octal
$u1127157$	27157	110101000010101	$6A15$	65025

In case if you don't have a UID please contact TAs.

Endianness. Endianness is the ordering or sequencing of bytes of a word in computer memory storage. Endianness is primarily expressed as big-endian or little-endian. Here is a link to an interesting story about the origin of terms big-endian and little-endian: https://www.ling.upenn.edu/courses/Spring$_2003/ling538/Lecnotes/ADfn1.htm$

Question 3. i) Explain briefly the little-endian and big-endian byte orders for multi-byte values in computer memory systems. [Refer to Byte Order section on page number A-43 in text book] **(10 points)**

The little-endian byte order for multi-byte values in computer memory systems assigns the least significant byte to the smallest memory address available. The big-endian byte order is the opposite. It assigns the most significant byte to the smallest memory address available.

ii) The contents of memory locations 200-203 and 400-403 are shown below. Assuming that the memory locations 200-203 are used to represent a 4-bytes word A and locations 400-403 are used for B, calculate the value of A and B in both the big-endian and little-endian order. (Note: the values must be represented in decimal.) [Refer to MIPS ISA II Lecture Slides 13 - 14] **(16 points)**

First, get the big-endian order and change the value to decimal form:
$5fcdb281_{hex} = 1607316097_{dec}$ this is the contents of the memory for A in big-endian order.

Now, get the little-endian order and change the value to decimal form:
$81b2cd5f_{hex} = 2175978847_{dec}$ this is the contents of the memory for A in little-endian order.

Get the big-endian order and change the value to decimal form:
$d36a0000_{hex} = 3546939392_{dec}$ this is the contents of the memory for B in big-endian order.

Get the little-endian order and change the value to decimal form:
$00006ad3_{hex} = 27347_{dec}$ this is the contents of the memory for B in little-endian order.

Contents of the memory for A (all values are in hex):

Address	200	201	202	203
Value	5f	cd	b2	81

Contents of the memory for B (all values are in hex):

Address	400	401	402	403
Value	d3	6a	00	00

Conversion to assembly. Below is an illustration of how to convert a high level language statement into assembly code with and without using temporary variables. Assume that the initial value of a and temporary variables t0 and t1 is zero.

a = b + c - d;

The assembly code using temporary variables:
```
add t0, b, c        # add b, c and store the result in t0.
sub f, t0, d        # subtract d from t0 and store the result in f.
```

The corresponding assembly code without using temporary variables:
```
add a, b, c        # add b, c and store the result in a.
sub a, a, d        # sub d from a and store result in a.
```

# - is a comment marker used in assembly code. Please use a comment marker along with each instruction to explain what it does as shown in the above sample programs.

Question 4. In this problem, we will practice converting high-level language code snippets to assembly. Translate the following high-level expressions to assembly using add and sub. Restrict the usage of temporary variables to t0 and t1 only. You can assume that the initial value of f and all the temporary variables is zero. [Refer to MIPS ISA II Lecture Slide 3]

(i) Translate f = (g - h) + (i - j) to assembly instructions with and without using temporary variables. **(15 points)**

With temporary variables:
sub t0, g, h
sub t1, i, j
add f, t0, t1

Without temporary variables:
sub f, g, h
add f, f, i
sub f, f, j

(ii) Translate f = (g) - (h + i) to assembly instructions with and without temporary variables. **(15 points)**

With temporary variables:
add t0, h, i
sub f, g, t0

Without temporary variables:
sub f, g, h
sub f, f, i
