CSE3666 Exam 1 Preparation

This is not exactly a practice exam. The actual exam will be different. However, most questions are similar to labs, homework, quizzes, iClicker questions, etc.

Please study homework, labs, lecture slides and quizzes, probably before trying to solve problems here.

The major topics covered in Exam 1 are posted on Discord.

Solutions will not be provided. However, students can write their solutions collaboratively.

Have trouble understanding stack and/or memory? Draw a figure.

Note that no calculator is allowed. If you feel that you are dealing with large numbers, probably you are not using a good method.

Another note on learning. Just looking at solutions and memorizing them are not learning. Try to explain the solutions to yourself or your friends.

1. Numbers

   0xAB

a) Consider the hexadecimal number as unsigned. Convert it to a decimal number.
b) Consider the bits denoted by the hex number as an 8-bit 2's complement numbers. Convert it to a decimal number.
c) Consider the bits denoted by the hex digits as an 8-bit 2's complement numbers. Negate the number, without converting to decimal numbers. Write the result as 8 bits.
d) Find the sum of the number (0xAB) and 0xF0. Represent the result as 8 bits.

2. Operations

a) Assume register s1 is 0x8642ACD0. Find out the bits in the destination register. Show the results as 8 hex digits.

```
ADDI    t0, s1, 0xFFFFF888
ANDI    t0, s1, 0xFFFFF888
ORI     t0, s1, 0xFFFFF888
XORI    t0, s1, 0xFFFFF888
SLLI    t0, s1, 16
SRAI    t0, s1, 16
SRLI    t0, s1, 16
```

b) The memory has the following bytes. Find out the bits in the destination register. Show the results as 8 hex digits.

Address	1000	1001	1002	1003
Value	0xAA	0xBB	0xCC	0xDD

```
lb          t4, 1002(x0)
lbu         t4, 1002(x0)
lh          t4, 1002(x0)
lhu         t4, 1002(x0)
lw          t5, 1000(x0)
```

c) Describe how a RSIC-V processor execute the following instructions.

```
ADD, ADDI, LW, SW, JAL, JALR, BEQ, LUI
```

d) How are the following pseudoinstruction are supported by RISC-V?

```
NOT, LI, JR
```

e) Count the number of instructions executed in a loop. See HW questions and examples in slides.

f) Give examples where BLT and BLTU behave the same (and differently). For example, if s1 = *x*, s2 = *y*, BLT and BLTU will both taken (or not taken).

3. Encoding/decoding.

a) Find the machine code of given instructions. Similar to homework questions.

b) Given machine code, find the instructions. Similar to homework questions.

Assemble some code in RARS. Check if you can a) find the machine code correctly, and b) write instructions from machine code. Exercise with all instruction formats.

c) Explain why the rd field has 5 bits.

d) If all R-type instructions have the same opcode, how many different R-type instructions can be supported?

e) What is the limitation on the immediate in instructions like ADDI?

f) Are the following two instructions interchangeable? Explain your answer.

```
BEQ     x0, x0, L1
JAL     x0, L1
```

4. Coding.

a) Is the following code correct? If not, explain the mistakes and correct them.

```
#load 0x2468ACE0 into register t0
li     t0, 0x2468
addi   t0, x0, 0xFFFFACE0
```

b) Write two instructions to jump to address `0x2468ACE0`. The two instructions work no matter where you place them.

c) Operation on arrays. For example, write a function that copies a word array to another word array.

d) Operation on strings. For example, write a function that convert a string to uppercase.

e) Save/restore registers on the stack. Save and restore registers ra, s1, s2 on/from the stack.

f) Allocate storage on the stack. Allocate two arrays on the stack. Save the starting address of w and c in registers s3 and s4, respectively. Draw a diagram that shows the layout of the stack after the space is allocated.

```
int     w[100];
char    c[128];
```

g) Call functions. For example, translate the following C code to RISC-V assembly code.

```
int foo(int a[])
{
     int   i;
     for (i = 0; i < 4096; i ++)
          a[i] = bar(&a[i]) + baz(a[i]+1);
     return;
```

```
}
```

Skeleton code of mergesort is posted on Discord. Try to complete the code. Students can come up solutions collaboratively.

Practice number conversions on the following site:

Binary Numbers (zhijieshi.github.io)

More materials are on the following pages.

cse3666/risc-v-idioms.md at master · zhijieshi/cse3666 (github.com)

cse3666/problems.md at master · zhijieshi/cse3666 (github.com)