

**NYU Tandon School of Engineering**

**Fall 2025, ECE 6913**

**Homework Assignment 2**

---

*Instructor: Azeez Bhavnagarwala, email: [ajb20@nyu.edu](mailto:ajb20@nyu.edu)*

*Course Assistant Office Hour Schedule*

On Zoom: 9:30AM – 11AM Monday, Tuesday, Wednesday, Thursday

Join Zoom: URL: <https://nyu.zoom.us/j/99480718922>

Abhishek Koti, [ak11546@nyu.edu](mailto:ak11546@nyu.edu)

Riya Shah, [rs9177@nyu.edu](mailto:rs9177@nyu.edu)

Vishwajeet Kulkarni, [vk2630@nyu.edu](mailto:vk2630@nyu.edu)

Shreyas Kootiganahalli, [sk12200@nyu.edu](mailto:sk12200@nyu.edu)

Please enter your responses in this Word document after you download it from NYU Classes.  
*Please use the Brightspace portal to upload your completed HW.*

---

- In RISC-V, only load and store instructions access memory locations
- These instructions must follow a ‘format’ to access memory
- Assume a 32-bit machine in all problems unless asked to assume otherwise

**Problem 1:**

Assume address in memory of ‘A[0]’, ‘B[0]’ and ‘C[0]’) are stored in Registers x27, x30, x31. Assume values of variables f, g, h, i, and j are assigned to registers x5, x6, x7, x28, x29 respectively

*Write down RISC V Instruction(s) to*

- (a) Load Register x5 with content of A[10]
- (b) Store contents of Register x5 into A[17]
- (c) add 2 operands: one in x5 - a register, the other in Register x6. Assume result of operation to be stored in register x7
- (d) copy contents at one memory location to another: C[g] = A[i+j+31]
- (e) implement in RISC V these line of code in C:
  - (i) f = g - A[B[9]]

- (ii)  $f = g - A[C[8] + B[4]]$
- (iii)  $A[i] = B[2i+1], C[i] = B[2i]$
- (iv)  $A[i] = 4B[i-1] + 4C[i+1]$
- (v)  $f = g - A[C[4] + B[12]]$

**Problem 2:**

Assume the following register contents:

$x5 = 0x00000000AAAAAAA, x6 = 0x1234567812345678$

- a. For the register values shown above, what is the value of  $x7$  for the following sequence of instructions?

```
srlt x7, x5, 16
addi x7, x7, -128
srai x7, x7, 2
andi x7, x7, x6
```

- b. For the register values shown above, what is the value of  $x7$  for the following sequence of instructions?

```
slli x7, x6, 4
```

- c. For the register values shown above, what is the value of  $x7$  for the following sequence of instructions?

```
srlt x7, x5, 3
andi x7, x7, 0xFEEF
```

### **Problem 3:**

For each RISC-V instruction below, identify the instruction format and show, wherever applicable, the value of the opcode (`op`), source register (`rs1`), source register (`rs2`), destination register (`rd`), immediate (`imm`), `func3`, `func7` fields. Also provide the 8 hex char (or 32 bit) instruction for each of the instructions below

```
add x5, x6, x7  
addi x8, x5, 512  
ld x3, 128(x27)  
sd x3, 256(x28)  
beq x5, x6 ELSE #ELSE is the label of an instruction 16 bytes larger  
#than the current content of PC  
add x3, x0, x0  
auipc x3, FFEFA  
jal x3 ELSE
```

### **Problem 4:**

**(a)** For the following C statement, write a minimal sequence of RISC-V assembly instructions that performs the identical operation. Assume `x5 = A`, and `x11` is the base address of C.

```
A = C[0] << 16;
```

**(b)** Find the shortest sequence of RISC-V instructions that extracts `bits 12 down to 7` from register `x3` and uses the value of this field to replace `bits 28 down to 23` in register `x4` without changing the other bits of registers `x3` or `x4`. (Be sure to test your code using `x3 = 0` and `x4 = 0xfffffffffffffff`. Doing so may reveal a common oversight.)

**(c)** Provide a minimal set of RISC-V instructions that may be used to implement the following pseudoinstruction:

```
not x5, x6 // bit-wise invert
```

*[Hint: note that there is no ‘not’ instruction in RISCV. However, an XOR immediate instruction could be used]*

### **Problem 5:**

Suppose the program counter (PC) is set to `0x60000000hex`.

- a. What range of addresses can be reached using the RISC-V *jump-and-link* (`jal`) instruction?  
(In other words, what is the set of possible values for the PC after the jump instruction executes?)
- b. What range of addresses can be reached using the RISC-V *branch if equal* (`beq`) instruction?  
(In other words, what is the set of possible values for the PC after the branch instruction executes?)

### **Problem 6:**

Assume that the register `x6` is initialized to the value 10. What is the final value in register `x5` assuming the `x5` is initially zero?

```
LOOP:      beq x6, x0, DONE
           addi x6, x6, -1
           addi x5, x5, 2
           jal x0, LOOP
```

DONE:

- a. For the loop above, write the equivalent C code. Assume that the registers `x5` and `x6` are integers `acc` and `i`, respectively.
- b. For the loop written in RISC-V assembly above, assume that the register `x6` is initialized to the value N. How many RISC-V instructions are executed?
- c. For the loop written in RISC-V assembly above, replace the instruction “`beq x6, x0, DONE`” with the instruction “`blt x6, x0, DONE`” and write the equivalent C code.

### **Problem 7:**

- a. Translate the following C code to RISC-V assembly code. Use a minimum number of instructions. Assume that the values of `a`, `b`, `i`, and `j` are in registers `x5`, `x6`, `x7`, and `x29`, respectively. Also, assume that register `x10` holds the base address of the array `D`.

```
for(i=0; i<a; i++)
    for(j=0; j<b; j++)
        D[4*j] = i + j;
```

- b. How many RISC-V instructions does it take to implement the C code from 7a. above? If the variables `a` and `b` are initialized to `10` and `1` and all elements of `D` are initially 0, what is the total number of RISC-V instructions executed to complete the loop?

### **Problem 8:**

Consider the following code:

```
lb x6, 0(x7)  
sd x6, 8(x7)
```

Assume that the register x7 contains the address **0x10000000** and the data at address is **0x1122334455667788**.

- a. What value is stored in **0x10000007** on a big-endian machine?
- b. What value is stored in **0x10000007** on a little-endian machine?

### **Problem 9:**

Write the RISC-V assembly code that creates the 64-bit constant **0x1234567812345678<sub>hex</sub>** and stores that value to register **x10**.

**Problem 10:** Assume that **x5** holds the value **128<sub>10</sub>**.

- a. For the instruction **add x30, x5, x6**, what is the range(s) of values for **x6** that would result in overflow?
- b. For the instruction **sub x30, x5, x6**, what is the range(s) of values for **x6** that would result in overflow?
- c. For the instruction **sub x30, x6, x5**, what is the range(s) of values for **x6** that would result in overflow?

### **Problem 10:**

Please read the ISSCC 2014 Keynote Publication by Professor Mark Horowitz “*Computing’s Energy Problem (and what we can do about it)*” [1]

1.1 How does Technology Scaling decrease the cost of Computing? How do reductions in the cost of manufacturing a transistor enable widespread use of computing devices?

1.2 Why did scaling processor clock frequency become more difficult in the last 20 years? How did Power Density/Dissipation become the primary constraint on server CPU performance?

1.3 Why is Moore’s Law slowing down? Why did Dennard Scaling end?

1.4 What component of energy consumption by Memory (in General Purpose Computing processors) is substantial ?

1.5 What solutions to Computing’s Energy Problem does Professor Mark Horowitz’s envision?

### **Problem 11:**

This assignment requires you to review 2 references on RISCV beginning with a summary transcript [2] of the Debate on Proprietary Vs Open-Source Instruction Sets at the 4<sup>th</sup> Workshop on Computer Architecture Research Directions, June 2015 sponsored by the ACM.

This Debate between Professor David Patterson (author of the textbook you are using) and Dave Christie of AMD highlights all of the key technical and business arguments for and against an Open-Source ISA such as RISC V as of 2015 (the same year the RISC V Foundation was established). A Technical Report from EECS UC Berkeley highlights the technical reasons for Open ISAs [3] providing a more detailed discussion on the advantages offered by open-source ISAs

- (1) Articulate *your* views on the topics debated in [2]. Justify your views.
- (2) Review and summarize technical reasons for Open-Source ISAs in [3].

### **References**

- [1] M Horowitz, “*Computing’s Energy Problem (and what we can do about it)*” Plenary Session 1.1, 2014 ISSCC Digest of Tech. papers, Feb 2014 [[PDF attached](#)]
- [2] M Hill et al, “Proprietary Versus Open Instruction Sets” 4th Workshop on Computer Architecture Research Directions, June 2015, ACM [[PDF attached](#)]
- [3] K Asanovic et al, “Instruction Sets Should Be Free: The Case For RISC-V”, EECS, University of California Berkeley, Technical Report No. UCB/EECS-2014-146, Aug 6 2014 [[PDF attached](#)]