COEN 210: Project, Max Version
Design a Pipelined CPU

Each group should have three students.

In this project, you will design a 32-bit pipelined CPU for the given SCU Instruction Set Architecture (SCU ISA). The SCU ISA is described below.

- Register file size: 64 registers, each register has 32 bits.
- PC: 32 bits
- 2's complement is assumed.
- Instruction format: Each instruction is 32-bit wide, and consists of five fields: `opcode, rd, rs, rt`, and `unused`. The format is as follows.

`Opcode (4 bits)`	`rd (6 bits)`	`rs (6 bits)`	`rt (6 bits)`	`unused(10 bits)`

The 13 instructions are defined below.

Instruction	Symbol	Opcode	rd	rs	rt	Function
No operation	**NOP**	**0000**	×	×	×	**No operation**
Save PC	**SVPC rd, y**	**1111**	**rd**	**y**		**xrd ← PC+y**
Load	**LD rd, rs**	**1110**	**rd**	**rs**	×	**xrd ← M[xrs]**
Store	**ST rt, rs**	**0011**	×	**rs**	**rt**	**M[xrs] ← xrt**
Add	**ADD rd, rs, rt**	**0100**	**rd**	**rs**	**rt**	**xrd ← xrs + x rt**
Increment	**INC rd, rs, y**	**0101**	**rd**	**rs**	**y**	**xrd ← xrs + y**
Negate	**NEG rd, rs**	**0110**	**rd**	**rs**	×	**x rd ← - xrs**
Subtract	**SUB rd, rs, rt**	**0111**	**rd**	**rs**	**rt**	**x rd ← xrs – x rt**
Jump	**J rs**	**1000**	×	**rs**	×	**PC ← xrs**
Branch if zero	**BRZ rs**	**1001**	×	**rs**	×	**PC ← xrs, if Z = 1**
Jump memory	**JM rs**	**1010**	×	**rs**	×	**PC ← M[xrs]**
Branch if negative	**BRN rs**	**1011**	×	**rs**	×	**PC ← xrs, if N = 1**
MAX	**MAX, rd, rs, rt**	**0001**	**rd**	**rs**	**rt**	**See ***

*****xrd = Max{memory[xrs], memory[xrs + 1], …, memory[xrs + xrt -1]},
×**:** don't care

Also, use the instructions in the SCU ISA to write two versions of assembly program of finding the maximum in *n* numbers described below.

(1) MAX = max $\{a_1, a_2, a_3, \dots, a_{n-1}, a_n\}$

The first version (software loop) does not use the MAX instruction and the second one (hardware loop) uses the MAX instruction.

You can create your new instructions to make the above coding easier. You may not need five stages in your pipeline.

When you analyze the cycle time, you can use the following delay data: delay of memory (I and D memory): 3ns., delay of register file: 2ns., delay of ALU (adders): 3ns. Ignore the delays of all other components.

Use the ALU attached to the last page.

The last instruction MAX is optional.

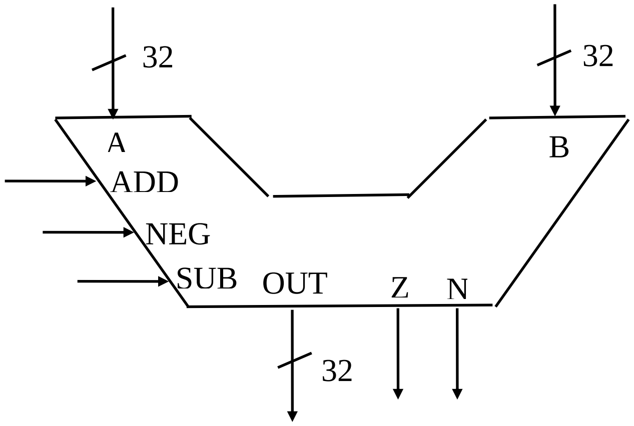
Submission (One submission per group)**:**

1. The two versions of your assembly codes for (1) in week 4 (20%). Use the given 13 instructions. **No label** is allowed in your code. Please upload your solution to Camino.
2. Your datapath and control (30%) in week 8. Upload your datapath to Camino.
3. Report (50%): Please submit a written report, including the following
    - Table of contents.
    - Assembly code.
    - Datapath and control.
    - Emulation of the following RISC-V instructions: `add, sub, addi, lw, sw, and, or, nor, andi, beq, jal, jalr`. For each RISC-V instruction mentioned, give the equivalent instruction or a sequence of instructions in SCU ISA.
    - Calculation of your cycle time, CPI, instruction count of the benchmark and the total execution time. Your design should minimize the execution time of the given benchmark.

Presentation: The presentation is at the **last lecture**. Each group will present its design for 20 minutes. The presentation takes 5 points out of 15 of the project.

Copying from projects of previous and current classes is defined as **cheating**.

ALU Block Diagram



ALU Truth Table

ADD	NEG	SUB	OUT	Operation
1	0	0	B+A	add
0	1	0	-B	2's complement
0	0	1	B-A	subtract
0	0	0	don't care	no operation
1	1	1	A	Pass A

Z=1 if and only if OUT=0
N=1 if and only if OUT is negative.