WELCOME

CSC258 LAB 1

Jan 16, 2023

Table of contents

TA Introductions

Laura Madrid

- 4th year Computer Science Specialist
- I like robotics , mobile app dev and I like to program in general
- I hope that you all really enjoy these labs and give you an appreciation for those that coded in assembly and paved the way 🥺🙏🏻

Yujie Wu

- 2nd year Master of Enginering (former cs specialist at UofT)
- Research interest: applied machine learning
- Hope you enjoy CSC258, as it really provides you with an alternative option for your future career, rather than just full stack developer

The DEMO

LET’S WALK THROUGH THE SET UP !!!

What you need:

01 Lab 1 folder

- Go to Quercus -> Weekly Schedule and Content -> Lab 1 -> download folder
- Have **lab01.pdf** for this demo & the activity

02 Ripes

- We need a RISC-V machine simulator …. and that's what Ripes is for ! This is where you will be running your assembly code and see what it does!
- Download from : *https://github.com/mortbopet/Ripes/releases/tag/v2.2.5*

03 RISC-V Reference

- Keeping these accesible during the lab will be super helpful, to remind you of the "commands" as you code in assembly (only need **32-bit instructions**)
- Reference card : *https://bit.ly/3eWFQ6V*
- Pseudoinstructions, good resource: *https://bit.ly/3pYgD2e*

Assembly Program:

Every assembly program we write will look very similar. Here is an outline of a typical program:

```
.data
# Add your constant and variable declarations here.
.globl main
.text

main:
  # Add your program code here.
  li a7, 10                # "Exit" is syscall 10. The next line will invoke a
                           # system call based on the value in a7.
  ecall                    # Always end your program with an exit.
```

beginning of the data section and the declaration

indicates the start of code.

Code needs to be within the **main:** section

Ripes Basics - Layout

Ripes File Edit View Help Fri Mar 4 12:15:20 PM

Ripes

100 ms

Editor

Processor

Cache

Memory

I/O

Source code

Input type: Assembly C

Executable code View mode: Binary Disassembled

```
151
152 delayLoop:
153     ecall
154     sub t2, a0, t1
155     bgez t2, delayIfEnd
156     addi t2, t2, -1
157
158 delayIfEnd:
159     bltu t2, t0, delayLoop
160     jr ra
161
162 # Takes in a number in a0, and returns a (sort of) random number from 0 to
163 # this number (exclusive)
164 rand:
165     mv t0, a0
166     li a7, 30
167     ecall
168     remu a0, a0, t0
169     jr ra
170
171 # Takes in an RGB color in a0, an x-coordinate in a1, and a y-coordinate
172 # in a2. Then it sets the led at (x, y) to the given color.
173 setLED:
174     li t1, LED_MATRIX_0_WIDTH
175     mul t0, a2, t1
176     add t0, t0, a1
177     li t1, 4
178     mul t0, t0, t1
179     li t1, LED_MATRIX_0_BASE
180     add t0, t1, t0
181     sw a0, (0)t0
182     jr ra
183
184 # Polls the d-pad input until a button is pressed, then returns a number
185 # representing the button that was pressed in a0.
186 # The possible return values are:
187 # 0: UP
188 # 1: DOWN
189 # 2: LEFT
190 # 3: RIGHT
191 pollDpad:
192     mv a0, zero
193     li t1, 4
194 pollLoop:
195     bge a0, t1, pollLoopEnd
196     li t2, D_PAD_0_BASE
197     slli t3, a0, 2
198     add t2, t2, t3
199     lw t3, (0)t2
```

```
00000000 <main>:
   0:   00400893    addi x17 x0 4
   4:   10000517    auipc x10 0x10000
   8:   ffc50513    addi x10 x10 -4
   c:   00000073    ecall
  10:   00400893    addi x17 x0 4
  14:   10000517    auipc x10 0x10000
  18:   04750513    addi x10 x10 71
  1c:   00000073    ecall
  20:   00000097    auipc x1 0x0 <main>
  24:   0b0080e7    jalr x1 x1 176
  28:   00050293    addi x5 x10 0
  2c:   00400893    addi x17 x0 4
  30:   10000517    auipc x10 0x10000
  34:   04e50513    addi x10 x10 78
  38:   00000073    ecall
  3c:   00000097    auipc x1 0x0 <main>
  40:   094080e7    jalr x1 x1 148
  44:   00050313    addi x6 x10 0
  48:   02a28393    addi x7 x5 42
  4c:   40530e33    sub x28 x6 x5
  50:   02530eb3    mul x29 x6 x5
  54:   00400893    addi x17 x0 4
  58:   10000517    auipc x10 0x10000
  5c:   04950513    addi x10 x10 73
  60:   00000073    ecall
  64:   00100893    addi x17 x0 1
  68:   00038513    addi x10 x7 0
  6c:   00000073    ecall
  70:   00400893    addi x17 x0 4
  74:   10000517    auipc x10 0x10000
  78:   04750513    addi x10 x10 71
  7c:   00000073    ecall
  80:   00400893    addi x17 x0 4
  84:   10000517    auipc x10 0x10000
  88:   02750513    addi x10 x10 39
  8c:   00000073    ecall
  90:   00100893    addi x17 x0 1
  94:   000e0513    addi x10 x28 0
  98:   00000073    ecall
  9c:   00400893    addi x17 x0 4
  a0:   10000517    auipc x10 0x10000
  a4:   01b50513    addi x10 x10 27
  a8:   00000073    ecall
  ac:   00400893    addi x17 x0 4
  b0:   10000517    auipc x10 0x10000
  b4:   00450513    addi x10 x10 4
```

GPR

Name	Alias	Value
x1	ra	0x00000000
x2	sp	0x7ffffff0
x3	gp	0x10000000
x4	tp	0x00000000
x5	t0	0x00000000
x6	t1	0x00000000
x7	t2	0x00000000
x8	s0	0x00000000
x9	s1	0x00000000
x10	a0	0x00000000
x11	a1	0x00000000
x12	a2	0x00000000
x13	a3	0x00000000
x14	a4	0x00000000
x15	a5	0x00000000
x16	a6	0x00000000

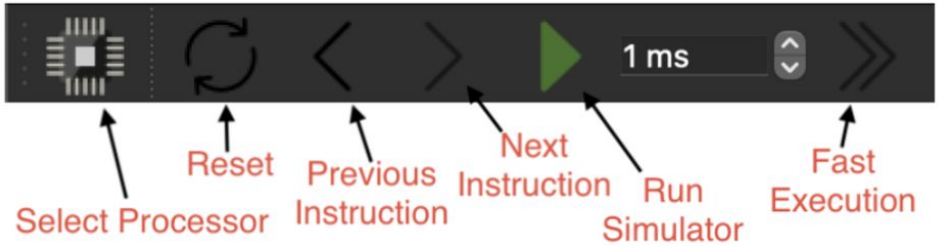
Display type: Hex

Processor: Single-cycle processor ISA: RV32IM

Ripes Basics - Layout

The application may be on dark mode, this is based on OS preferences

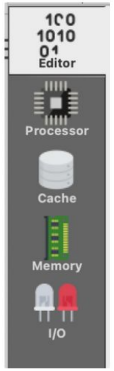
The menu at the top of the screen has buttons that pertain to building and modifying the project



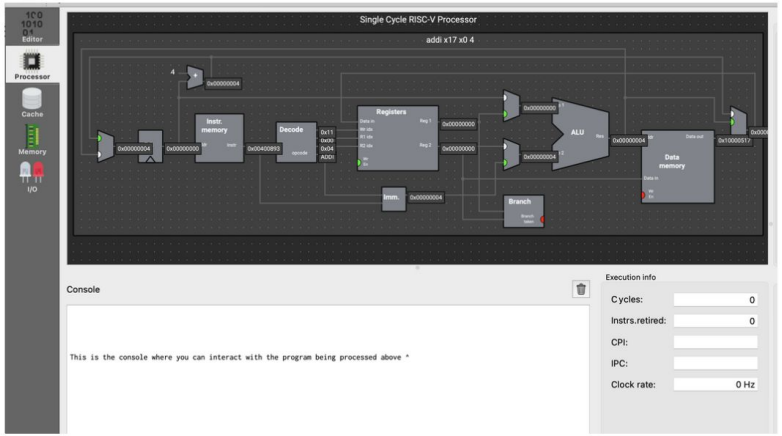
- **Select processor:** Where you can choose what processor to run the program on
- **Reset:** Resets the code that is run on ripes
- **Run Simulator:** Runs the program line by line
    - The next and previous instruction buttons are used to go to the previous line and next line respectively
- **Fast Execution:** Runs the program immediately

Ripes Basics - Layout

The Menu on the left-hand side of the screen pertains to the execution and the interface of the program:



- **Editor:** The tab where the code of the program is found
- **Processor:** The tab where you can visually see each instruction of the program get processed (at the top of the screen) & where you can interact with the program through the console (at the bottom of the screen)



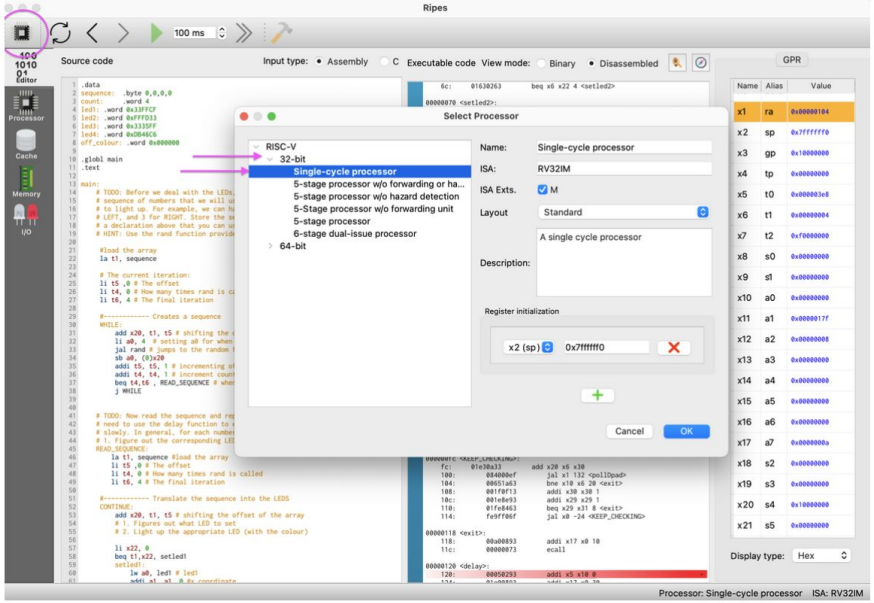
- **Cache & Memory:** Keep track of the components and memory of the program
- **I/O:** The tab where LED and d-pad controller interaction is found

Ripes Basics - Setup

Before running any code please do the following:

"Select Processor" and choose a 32-bit processor for now, we'll use the "Single-cycle processor".

- Click on the processor tab at the top menu of the screen (circled in magenta below)
  - Click on the **32-bit processor** menu and select ***Single-cycle processor***, then select OK at the bottom of the Select processor screen



Read the PDF for a more in-depth explanation!

The Activity

LET’S MAKE YOUR FIRST PROGRAM !!!

TODOs

01 Modify Sample.s

- Read through the handout and get familiar with the procedure if needed
- Finish the TODO parts in the code and ask the TA for any questions you have

02 Create “lab1a.s” and “lab1b.s”

- **lab1a.s**: Prompt the user for A and B, and output results A + 42 and B - A
- **lab1b.s**: Prompt the user for A, B and C and output A + B + C

03 Submit “lab1b.s” on Quercus

- Once you are done, show us and we can grade your completion of the lab
- If you’re unable to finish during lab, submit by **5 p.m. on Friday**