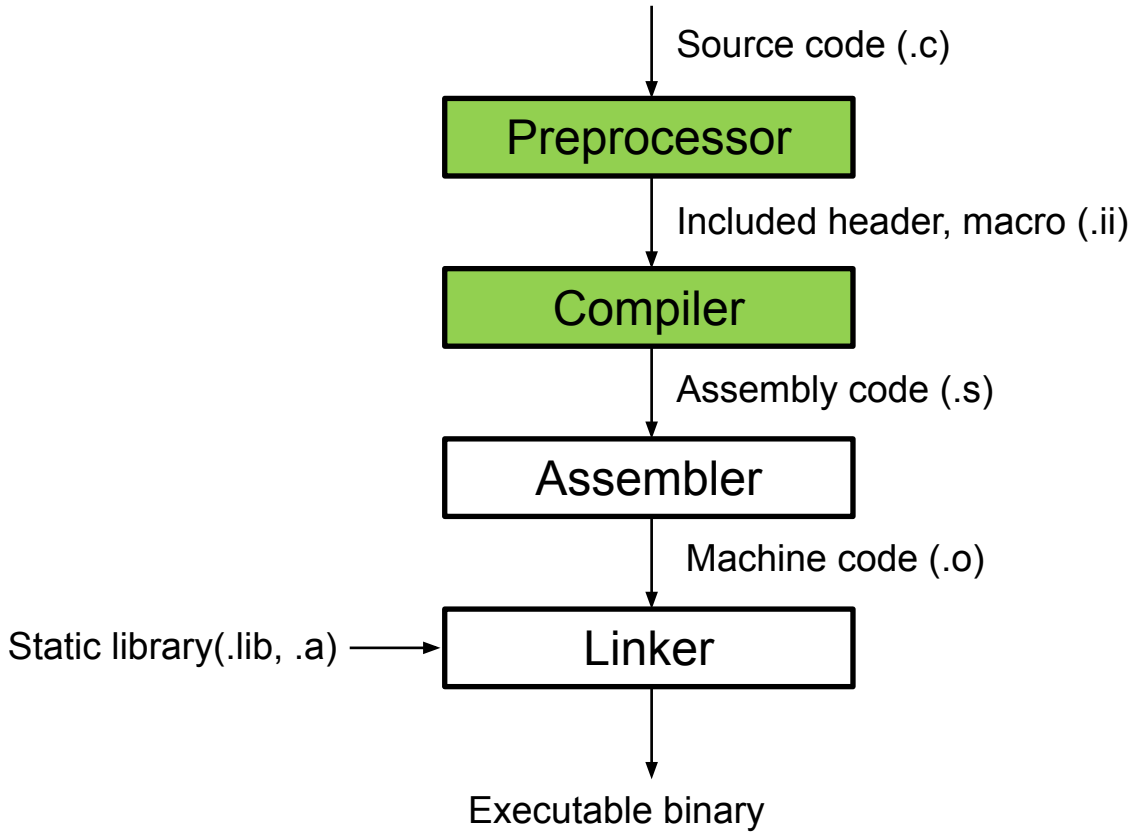
# Programming Assignment #1: Human Compiler

Prof. Jae W. Lee (jaewlee@snu.ac.kr)
Department of Computer Science and Engineering
Seoul National University

TA (snu-arc-uarch-ta@googlegroups.com)

# Goal of this project

- **You will compile C source code into 32-bit RISC-V assembly.**



# Experimental setup

- **You will use RISC-V ISA simulator on Linux: Two options**
  - Refer to experimental setup slide
  - Option 1: Use VirtualBox (recommend)
    - download & install Virtual Box from https://www.virtualbox.org/wiki/Downloads
    - download container Image from https://drive.google.com/file/d/1a1CTfbebIOAFyXwykXw3OZtrGs8b-Gjr/view?usp=sharing and import
  - Option 2: Use your own Linux box
    - CAVEAT: Grading will be done on our VM
    - download file from https://drive.google.com/file/d/1IGsdbDlsnaInFaK0oQDt8iiedYJwt3zY/view?usp=sharing
  - Option 3 : (hidden) Use Docker
  - Please refer to ‘experimental setup’ slides

# Experimental setup

- **github link**
  **https://github.com/SNU-ARC/2023_fall_comarch_PA1**

- **code download**
  **git clone https://github.com/SNU-ARC/2023_fall_comarch_PA1.git**

- **feel free to ask question about PA1 at git issue**
  **https://github.com/SNU-ARC/2023_fall_comarch_PA1/issues**
- **keep your repository private if you're going to fork it**

# Experimental setup

- **How to execute your code (for both Option 1 and 2):**

In directory

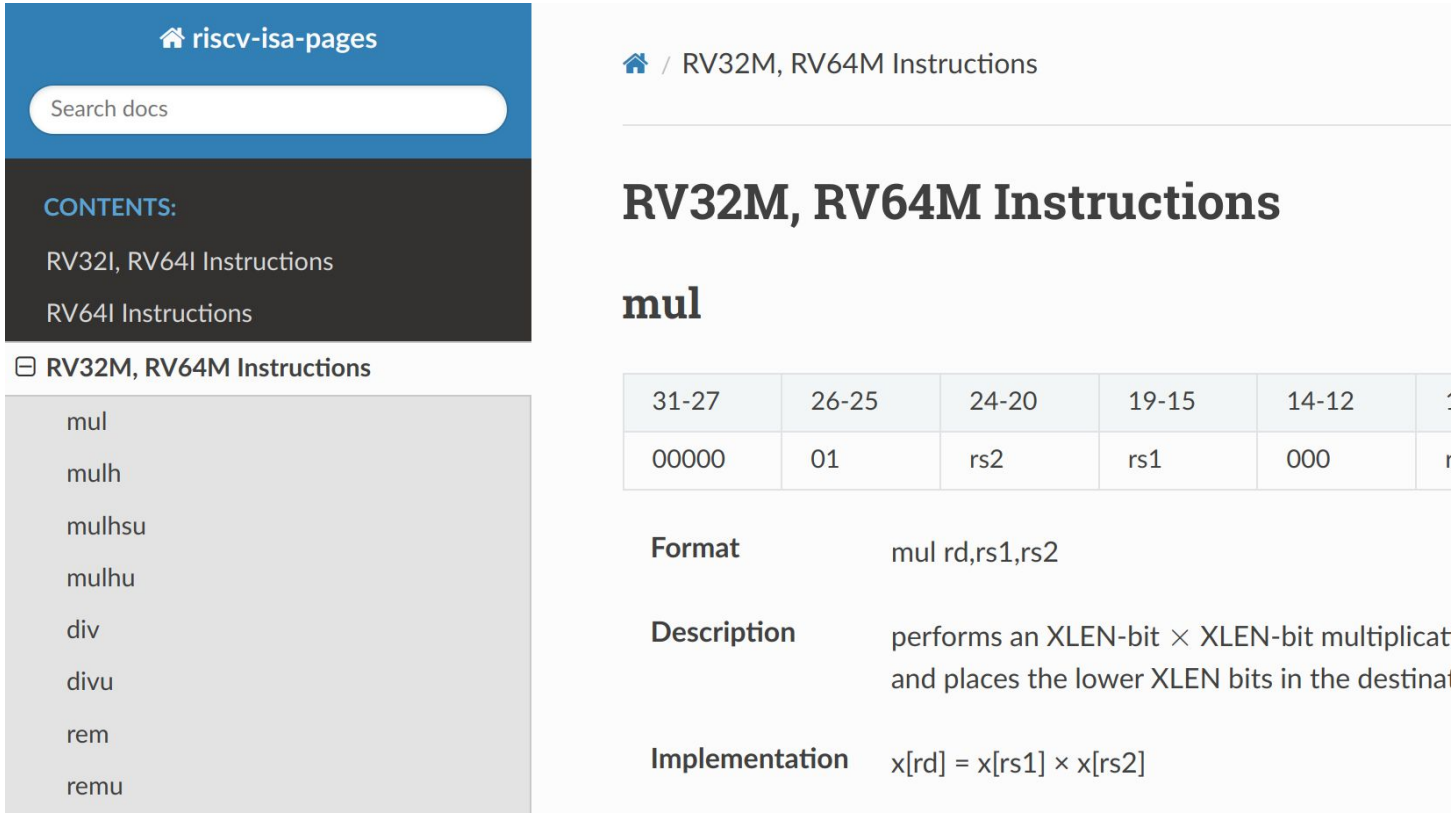
```
$> Make
$> sh run.sh
```

```
root@00150ad65406:/# cd /home/digitsum/
root@00150ad65406:/home/digitsum# make
riscv32-unknown-elf-gcc -Wall -Werror -std=c99 -c main.c -o main.o
riscv32-unknown-elf-gcc -c digitsum_asm.s -o digitsum_asm.o
riscv32-unknown-elf-gcc main.o digitsum_asm.o -o digitsum
root@00150ad65406:/home/digitsum# sh run.sh
bbl loader
digitsum of 123, 456 = 21
```

# References

Instructions of RV32I , RV32M is allowed

**https://msyksphinz-self.github.io/riscv-isadoc/html/rvi.html#ecall**



# Problem 1: Digit sum

- **Calculate the sum of digits of two integers.**
  - Digit sum of 15, 234 is 1+5+2+3+4 = 15
  - Write your code on `digitsum_asm.s`
  - Refer to `digitsum.c` (reference code) for algorithm.
  - Operands are stored at register a0, a1.
  - Store the answer to register a0 and return.
  - Execution:

```
$> sh run.sh
  or
$> spike --isa=RV32IM /opt/riscv/bin/pk ./digitsum [lhs] [rhs]
```

# Problem 2: Fibonacci sequence

- **Return value of the given number of Fibonacci sequence.**
  - Write your code on `fibonacci_asm.s`
  - Refer to `fibonacci.c` (reference code) for algorithm.
  - You should make your code with **recursion**.
  - The length of the sequence (n) is stored in register a0.
  - Store the answer to register a0 and return.
  - Execution:

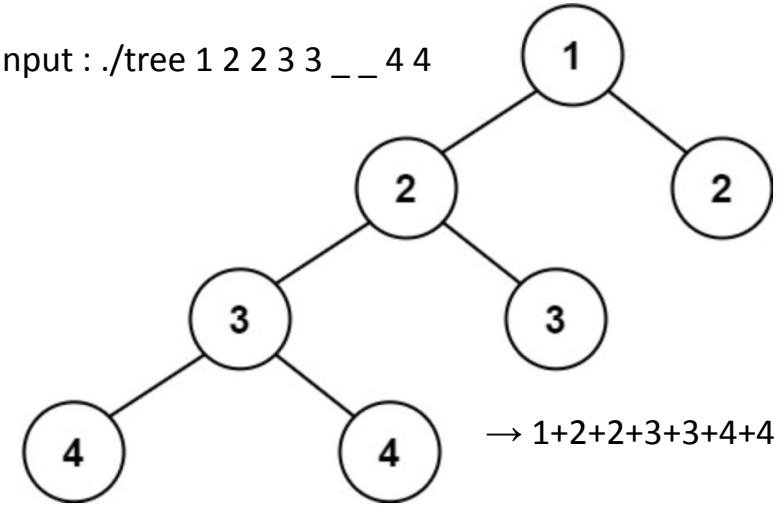
```
$> sh run.sh
  or
$> spike --isa=RV32IM /opt/riscv/bin/pk ./fibonacci [count]
```

# Problem 3: Tree sum

- **Calculate sum of Tree that is given.**
  - Input sequence will be made to binary tree. You can skip nodes with character '_'
  - Struct of tree node is inside of tree.h
  - Refer to main.c how tree is made.
  - Refer to tree.c for algorithm.
  - Number of tree node is maximum 15.
  - traverse all of the tree nodes once and calculate sum of all.

```
$> sh run.sh
```

input : ./tree 1 2 2 3 3 _ _ 4 4




→ 1+2+2+3+3+4+4 = 19

# Problem 3: Tree sum

- **How to make tree from inputs**
  - Binary tree for 15 entries
  - If input : ./tree 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 then figure 1

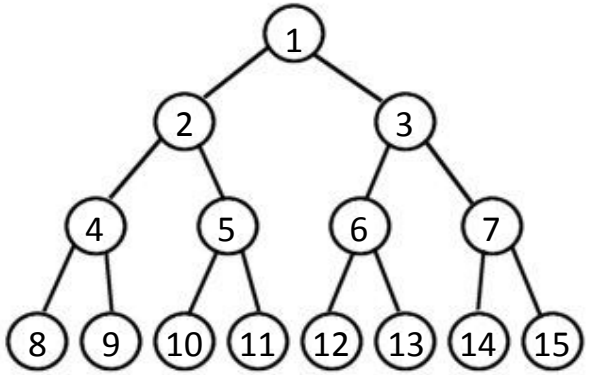



figure 1

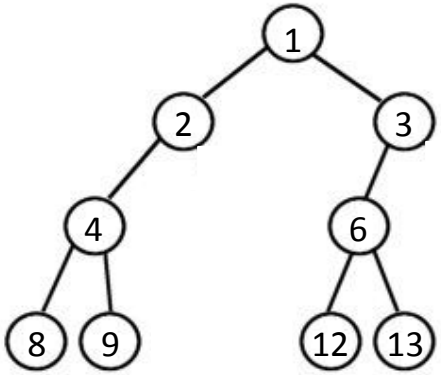



figure 2

  - Put '_' to skip nodes
  - If you want to erase '5 and childs, '7 and childs' like figure 2 then, input will be ./tree 1 2 3 4 _ 6 _ 8 9 _ _ 12 13

# Submission

- **Write-up**
  - Briefly describe your implementation (≤5 pages)
  - Filename: [student_id].pdf (example: 2023-12345.pdf)
  - **Please** submit it in **PDF** format. Other formats are not accepted.
- **Compress your source code and write-up into a single zip file**
  - Compress digitsum_asm.s, fibonacci_asm.s, tree_asm.s and your write-up
  - Filename should be [student_id].zip (example: 2023-12345.zip).
  - **Please** submit it in **ZIP** format. Other formats are not accepted.
- **Submission deadline: 2023. 9. 27 (Wed) 23:59**
  - Submit via ETL
- **For late submission**
  - 10% deduction every 24 hours
  - After next 120 hours: Submission not accepted