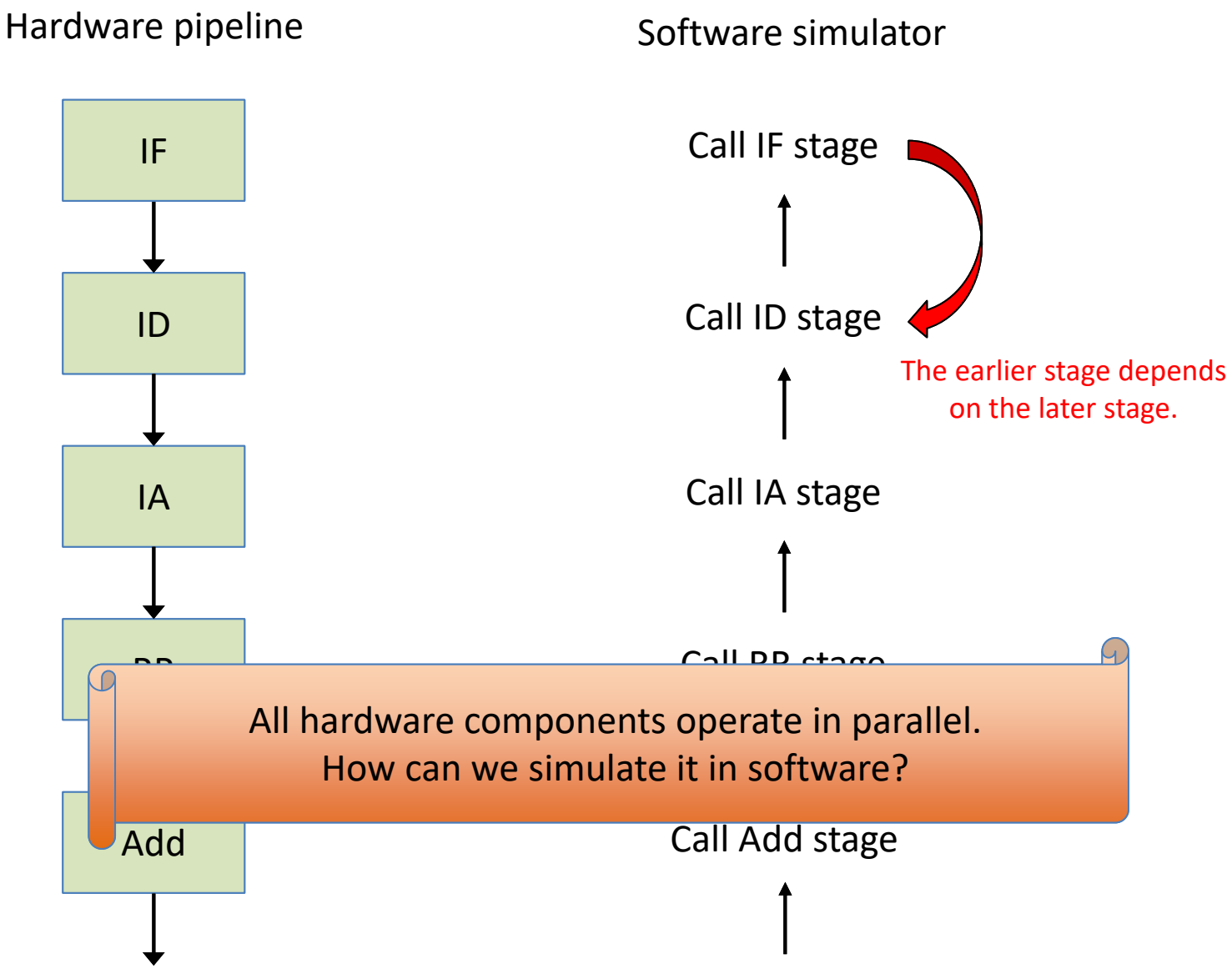
CS 520 Computer Architecture

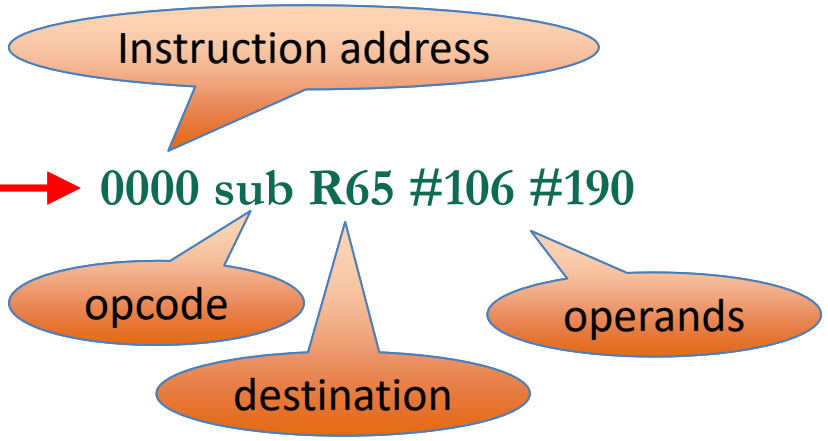
01. Project #1

Programming for CPU simulator



Instructions

0000 sub R65 #106 #190
0004 add R60 #120 #234
0008 mul R33 #132 #252
0012 set R34 #70
0016 set R28 #3
0020 div R54 #71 #146
0024 sub R2 #99 #42
0028 sub R104 #24 #123
0032 add R16 #101 #46
0036 div R106 #43 #218
0040 ret



- Each instruction is 4B
- The provided program files hold instructions in a string format
- Instructions locate in memory address range from 0 to 999

Memory Map (1)

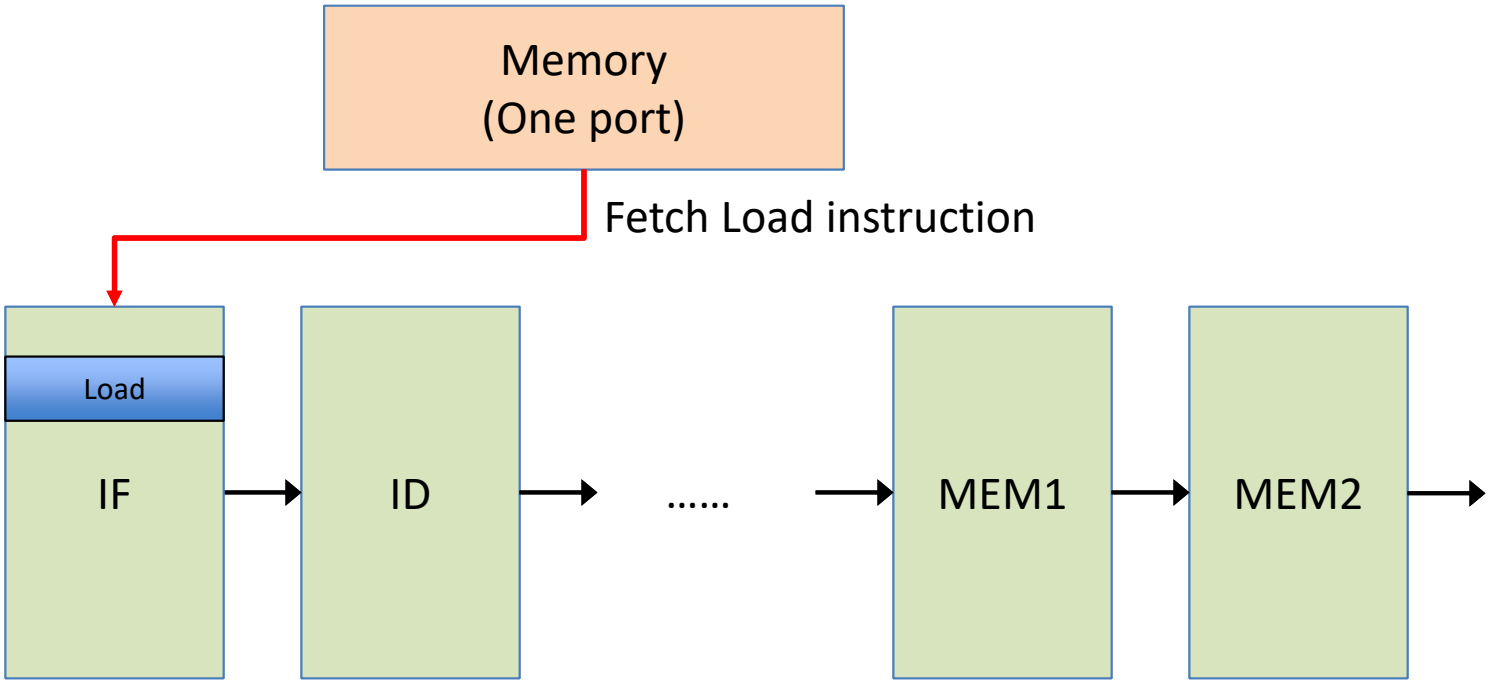

0
Text
1000
Data
65535
The first 1000B is reserved for programs (256 numbers).
Each number presents data in 4B memory.
The memory from 1000 to 65535 is reserved for data.

Memory Map (2)

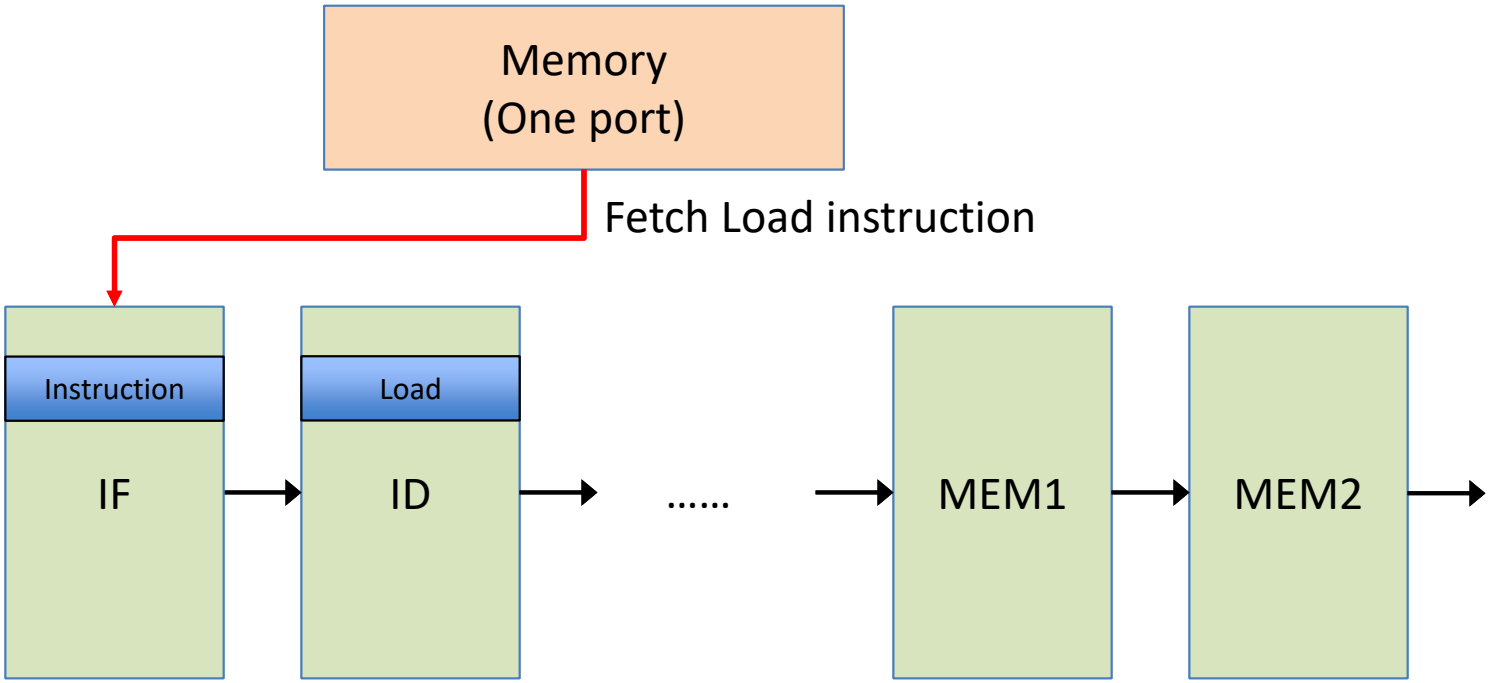
- **Memory map file**

- Each number presents data in 4B memory
  - Memory map size is 64KB, having 16,384 numbers.
  - Memory address is mapped to a specific location in the memory map
  - e.g., 2400 means the 600th number
- The provided memory map file is used by all programs commonly
  - Address range 0-999 is reserved for programs
- Format in the map file
  - All numbers are written in a string format
  - Your codes need to open the map file and read the number at a specific location
  - e.g, ld R95 #46580// load the data stored in 46580 into register Rx

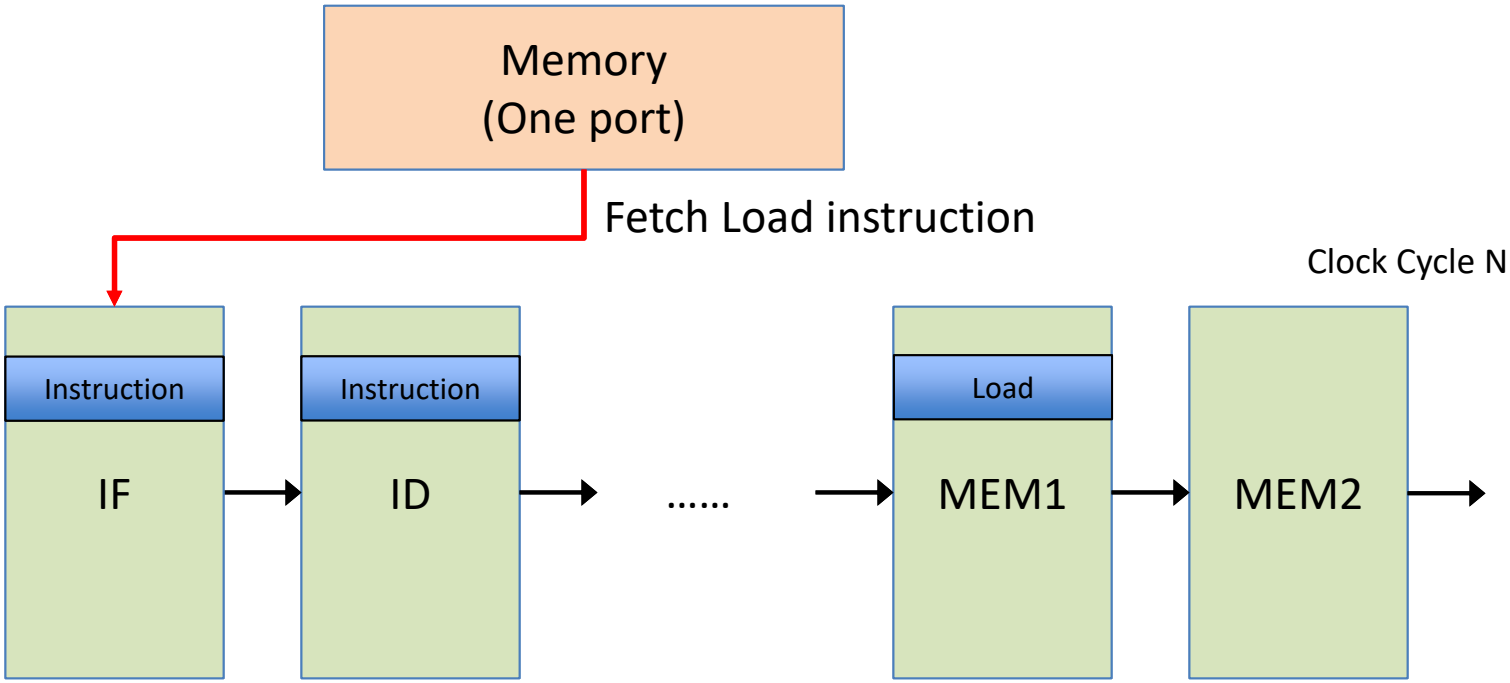
Structural Hazard (1)



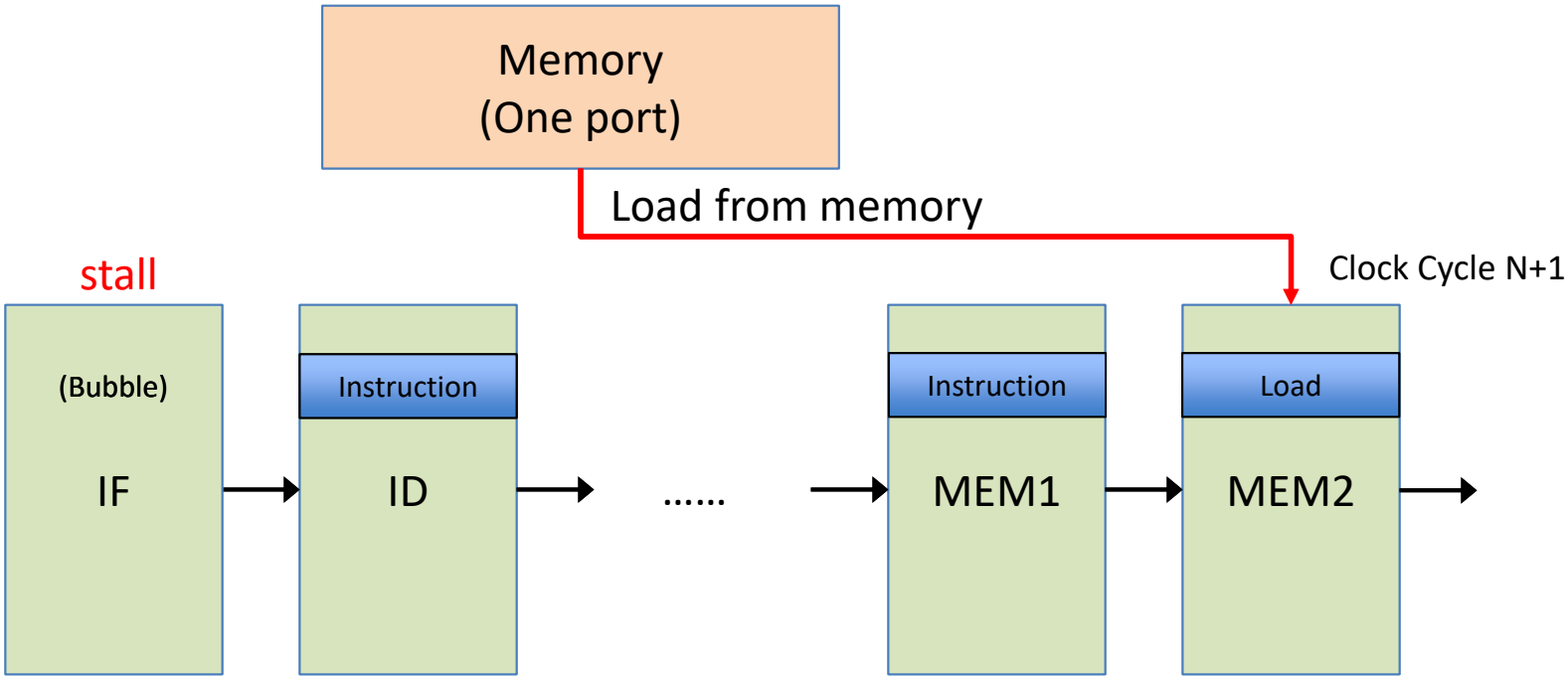
Structural Hazard (2)



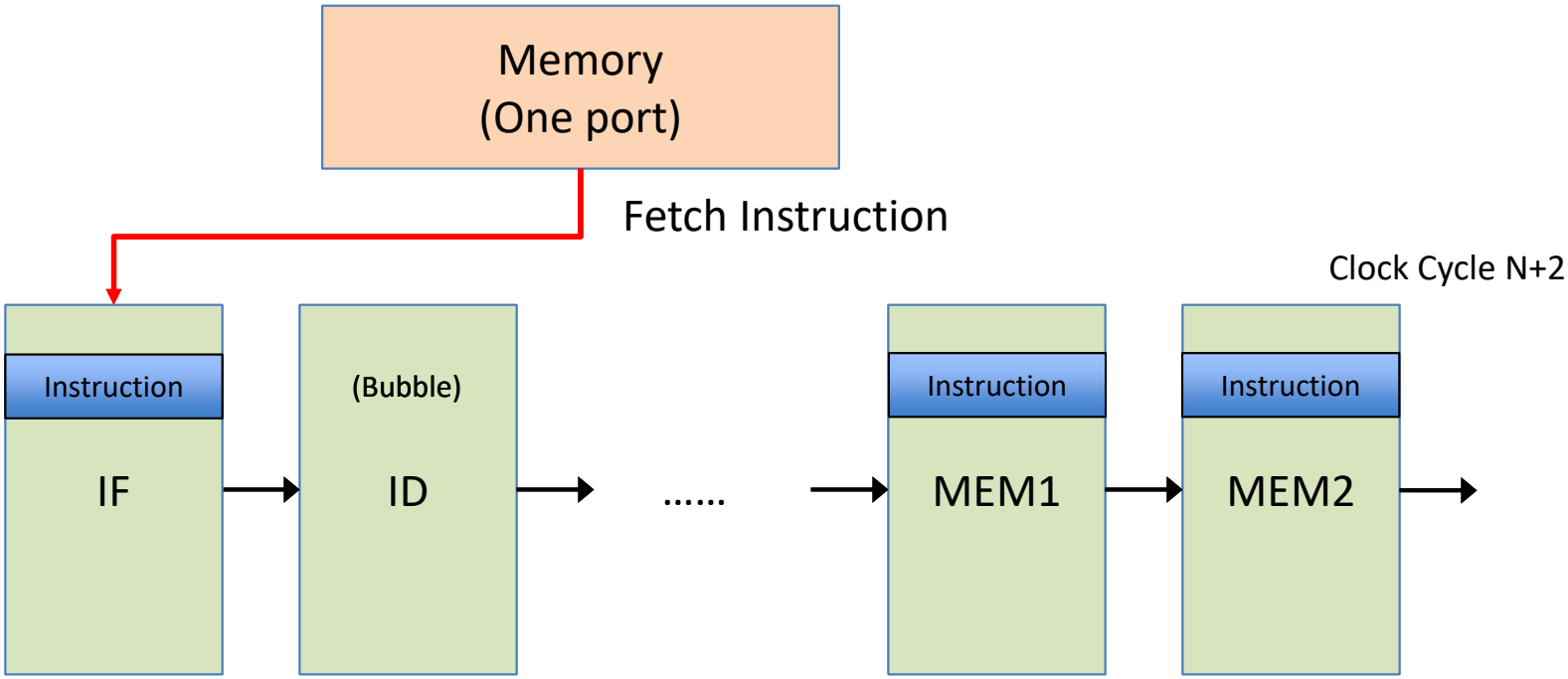
Structural Hazard (3)



Structural Hazard (4)



Structural Hazard (5)



Logs (1) – name_stats.txt

```
Stalled cycles due to structural hazard: 0
Total execution cycles: 21
Total instruction simulated: 11
IPC: 0.523810
```

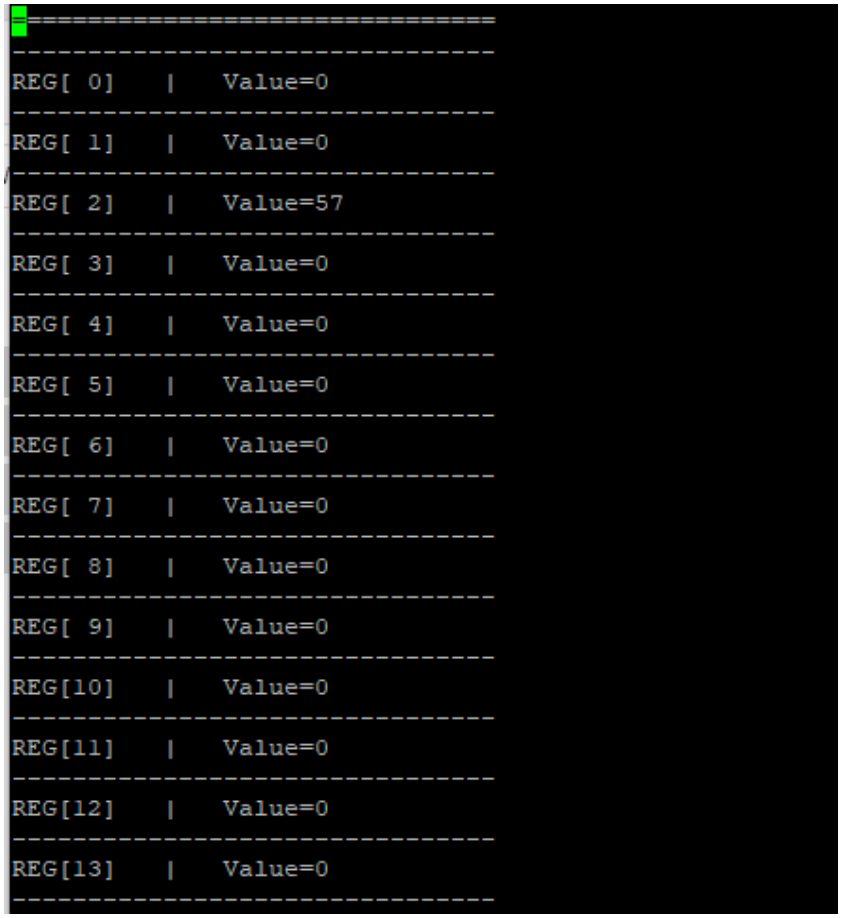
- _stats.txt shows the performance stats such as execution cycles and IPC

Logs (2) – name_pipeline.txt

```
================================
Clock Cycle #: 1
--------------------------------
IF              : 0000 sub R65 #106 #190
================================
Clock Cycle #: 2
--------------------------------
ID              : 0000 sub R65 #106 #190
IF              : 0004 add R60 #120 #234
================================
Clock Cycle #: 3
--------------------------------
IA              : 0000 sub R65 #106 #190
ID              : 0004 add R60 #120 #234
IF              : 0008 mul R33 #132 #252
================================
Clock Cycle #: 4
--------------------------------
RR              : 0000 sub R65 #106 #190
IA              : 0004 add R60 #120 #234
ID              : 0008 mul R33 #132 #252
IF              : 0012 set R34 #70
================================
Clock Cycle #: 5
--------------------------------
ADD             : 0000 sub R65 #106 #190
RR              : 0004 add R60 #120 #234
IA              : 0008 mul R33 #132 #252
ID              : 0012 set R34 #70
IF              : 0016 set R28 #3
```

- _pipeline.txt shows the instruction information in each stage at a specific clock cycle

Logs (3) – name_registerstatus.txt



- _registerstatus.txt shows the final register status

Logs (4) – name_result.txt

```
================================
Clock Cycle #: 1
--------------------------------
IF              : 0000 sub R65 #106 #190
=============== STATE OF ARCHITECTURAL REGISTER FILE ==========

|   REG[ 0]   |   Value=0   |   Status=valid   |
|   REG[ 1]   |   Value=0   |   Status=valid   |
|   REG[ 2]   |   Value=0   |   Status=valid   |
|   REG[ 3]   |   Value=0   |   Status=valid   |
|   REG[ 4]   |   Value=0   |   Status=valid   |
|   REG[ 5]   |   Value=0   |   Status=valid   |
|   REG[ 6]   |   Value=0   |   Status=valid   |
|   REG[ 7]   |   Value=0   |   Status=valid   |
|   REG[ 8]   |   Value=0   |   Status=valid   |
|   REG[ 9]   |   Value=0   |   Status=valid   |
|   REG[10]   |   Value=0   |   Status=valid   |
|   REG[11]   |   Value=0   |   Status=valid   |
|   REG[12]   |   Value=0   |   Status=valid   |
|   REG[13]   |   Value=0   |   Status=valid   |
|   REG[14]   |   Value=0   |   Status=valid   |
|   REG[15]   |   Value=0   |   Status=valid   |
|   REG[16]   |   Value=0   |   Status=valid   |
|   REG[17]   |   Value=0   |   Status=valid   |
|   REG[18]   |   Value=0   |   Status=valid   |
|   REG[19]   |   Value=0   |   Status=valid   |
|   REG[20]   |   Value=0   |   Status=valid   |
|   REG[21]   |   Value=0   |   Status=valid   |
|   REG[22]   |   Value=0   |   Status=valid   |
|   REG[23]   |   Value=0   |   Status=valid   |
```

- _result.txt shows all the information including the register status at each cycle