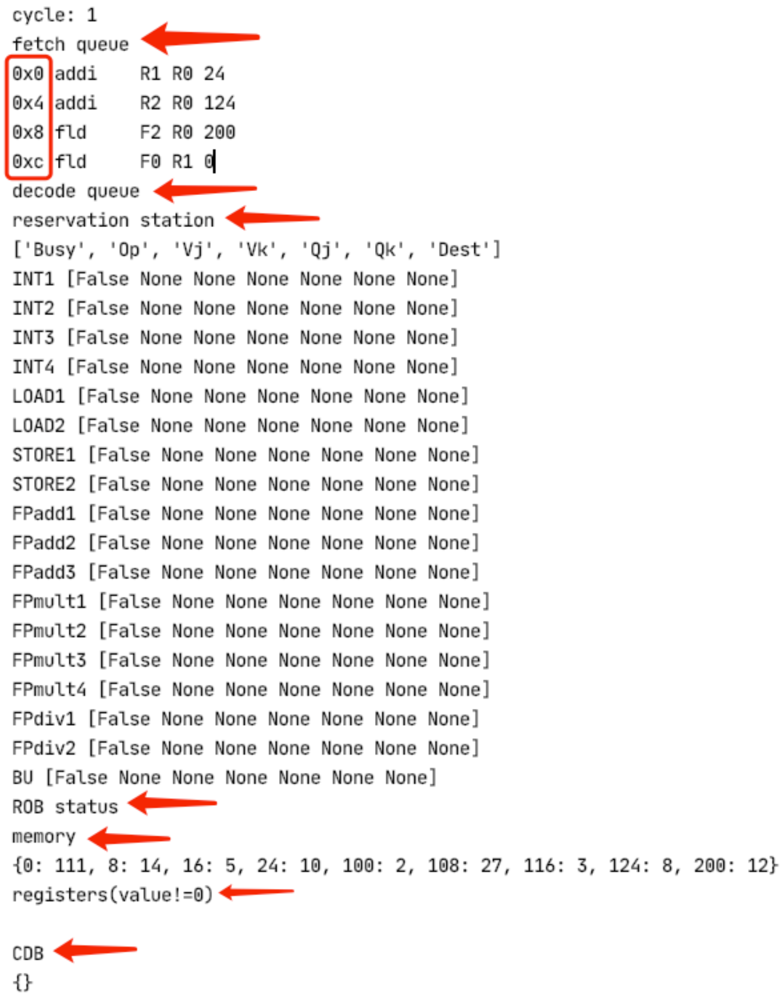
Course Project Report

Bangyan Li - BAL116

1. Design description

(1) Explanation of the output result



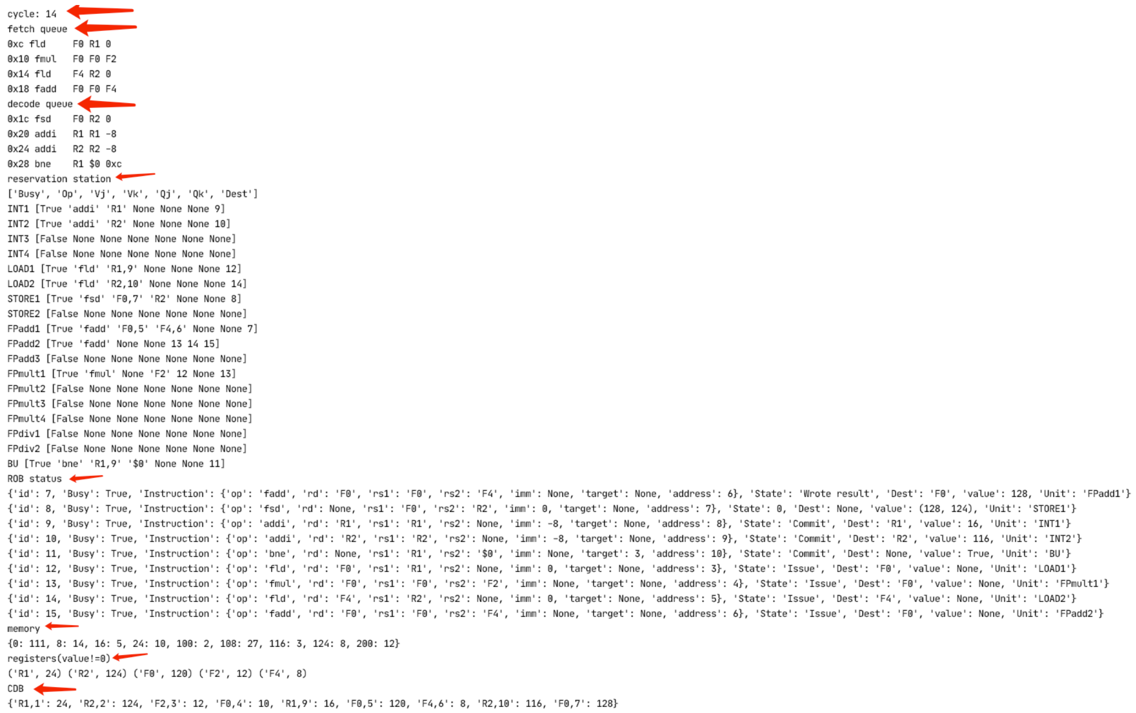
```
cycle: 1
fetch queue
0x0 addi    R1 R0 24
0x4 addi    R2 R0 124
0x8 fld     F2 R0 200
0xc fld     F0 R1 0
decode queue
reservation station
['Busy', 'Op', 'Vj', 'Vk', 'Qj', 'Qk', 'Dest']
INT1 [False None None None None None None]
INT2 [False None None None None None None]
INT3 [False None None None None None None]
INT4 [False None None None None None None]
LOAD1 [False None None None None None None]
LOAD2 [False None None None None None None]
STORE1 [False None None None None None None]
STORE2 [False None None None None None None]
FPadd1 [False None None None None None None]
FPadd2 [False None None None None None None]
FPadd3 [False None None None None None None]
FPmult1 [False None None None None None None]
FPmult2 [False None None None None None None]
FPmult3 [False None None None None None None]
FPmult4 [False None None None None None None]
FPdiv1 [False None None None None None None]
FPdiv2 [False None None None None None None]
BU [False None None None None None None]
ROB status
memory
{0: 111, 8: 14, 16: 5, 24: 10, 100: 2, 108: 27, 116: 3, 124: 8, 200: 12}
registers(value!=0)

CDB
{}
```

This figure illustrates the case of cycle=1 when the program starts running.

There are four instructions in the fetch queue, and the red box indicates the address of the instruction.

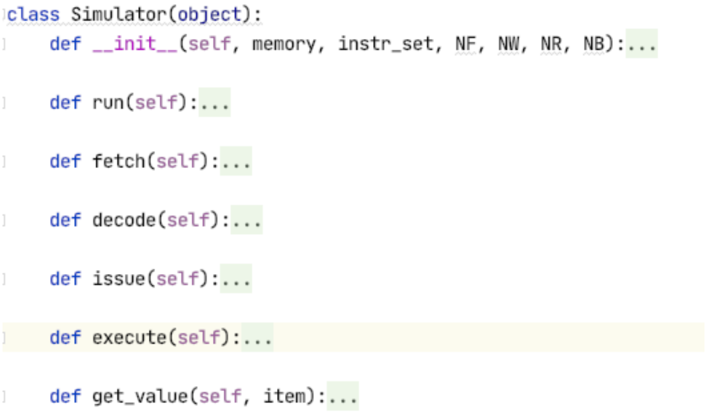
Because cycle=1, decode queue is empty, 'Busy' in reservation station is also false, ROB status is also empty, memory is read from memory content file. And the program only print registers whose value is not 0. The CDB is currently also empty.



```
cycle: 14
fetch queue
0xc fld     F0 R1 0
0x10 fmul   F0 F0 F2
0x14 fld    F4 R2 0
0x18 fadd   F0 F0 F4
decode queue
0x1c fsd    F0 R2 0
0x20 addi   R1 R1 -8
0x24 addi   R2 R2 -8
0x28 bne    R1 $0 0xc
reservation station
['Busy', 'Op', 'Vj', 'Vk', 'Qj', 'Qk', 'Dest']
INT1 [True 'addi' 'R1' None None None 9]
INT2 [True 'addi' 'R2' None None None 10]
INT3 [False None None None None None None]
INT4 [False None None None None None None]
LOAD1 [True 'fld' 'R1,9' None None None 12]
LOAD2 [True 'fld' 'R2,10' None None None 14]
STORE1 [True 'fsd' 'F0,7' 'R2' None None 8]
STORE2 [False None None None None None None]
FPadd1 [True 'fadd' 'F0,5' 'F4,6' None None 7]
FPadd2 [True 'fadd' None None 13 14 15]
FPadd3 [False None None None None None None]
FPmult1 [True 'fmul' None 'F2' 12 None 13]
FPmult2 [False None None None None None None]
FPmult3 [False None None None None None None]
FPmult4 [False None None None None None None]
FPdiv1 [False None None None None None None]
FPdiv2 [False None None None None None None]
BU [True 'bne' 'R1,9' '$0' None None 11]
ROB status
{'id': 7, 'Busy': True, 'Instruction': {'op': 'fadd', 'rd': 'F0', 'rs1': 'F0', 'rs2': 'F4', 'imm': None, 'target': None, 'address': 6}, 'State': 'Wrote result', 'Dest': 'F0', 'value': 128, 'Unit': 'FPadd1'}
{'id': 8, 'Busy': True, 'Instruction': {'op': 'fsd', 'rd': None, 'rs1': 'F0', 'rs2': 'R2', 'imm': 0, 'target': None, 'address': 7}, 'State': 0, 'Dest': None, 'value': (128, 124), 'Unit': 'STORE1'}
{'id': 9, 'Busy': True, 'Instruction': {'op': 'addi', 'rd': 'R1', 'rs1': 'R1', 'rs2': None, 'imm': -8, 'target': None, 'address': 8}, 'State': 'Commit', 'Dest': 'R1', 'value': 16, 'Unit': 'INT1'}
{'id': 10, 'Busy': True, 'Instruction': {'op': 'addi', 'rd': 'R2', 'rs1': 'R2', 'rs2': None, 'imm': -8, 'target': None, 'address': 9}, 'State': 'Commit', 'Dest': 'R2', 'value': 116, 'Unit': 'INT2'}
{'id': 11, 'Busy': True, 'Instruction': {'op': 'bne', 'rd': None, 'rs1': 'R1', 'rs2': '$0', 'imm': None, 'target': 3, 'address': 10}, 'State': 'Commit', 'Dest': None, 'value': True, 'Unit': 'BU'}
{'id': 12, 'Busy': True, 'Instruction': {'op': 'fld', 'rd': 'F0', 'rs1': 'R1', 'rs2': None, 'imm': 0, 'target': None, 'address': 3}, 'State': 'Issue', 'Dest': 'F0', 'value': None, 'Unit': 'LOAD1'}
{'id': 13, 'Busy': True, 'Instruction': {'op': 'fmul', 'rd': 'F0', 'rs1': 'F0', 'rs2': 'F2', 'imm': None, 'target': None, 'address': 4}, 'State': 'Issue', 'Dest': 'F0', 'value': None, 'Unit': 'FPmult1'}
{'id': 14, 'Busy': True, 'Instruction': {'op': 'fld', 'rd': 'F4', 'rs1': 'R2', 'rs2': None, 'imm': 0, 'target': None, 'address': 5}, 'State': 'Issue', 'Dest': 'F4', 'value': None, 'Unit': 'LOAD2'}
{'id': 15, 'Busy': True, 'Instruction': {'op': 'fadd', 'rd': 'F0', 'rs1': 'F0', 'rs2': 'F4', 'imm': None, 'target': None, 'address': 6}, 'State': 'Issue', 'Dest': 'F0', 'value': None, 'Unit': 'FPadd2'}
memory
{0: 111, 8: 14, 16: 5, 24: 10, 100: 2, 108: 27, 116: 3, 124: 8, 200: 12}
registers(value!=0)
('R1', 24) ('R2', 124) ('F0', 120) ('F2', 12) ('F4', 8)
CDB
{'R1,1': 24, 'R2,2': 124, 'F2,3': 12, 'F0,4': 10, 'R1,9': 16, 'F0,5': 120, 'F4,6': 8, 'R2,10': 116, 'F0,7': 128}
```

This figure illustrates the case of cycle=14 when the program is running

There are 4 instructions in the fetch queue and 4 instructions in the decode queue. The row where busy is true in the reservation station indicates that the function unit is occupied. ROB status will update the state of the item in order. In the common data bus. "R1, 1":24 means that the instruction with the ID of 1 modifies the value of R1 to 24. The purpose of adding the ID is to allow the instruction to get the value it depends on through the CDB.



```python
class Simulator(object):
    def __init__(self, memory, instr_set, NF, NW, NR, NB):...

    def run(self):...

    def fetch(self):...

    def decode(self):...

    def issue(self):...

    def execute(self):...

    def get_value(self, item):...
```

This is the definition of simulator. I created four methods of fetch, decode, issue, and execute for the simulator object to simulate the process. The state whose state is M will be processed in the execute function. And function get_value is for the calculation of the instruction result.

2. Comparative Analysis

When NF=4, NW=4, NR=16, NB=4, total cycle is 38,

1) When NF=4, NW=2, NR=16, NB=2, total cycle is 39,

Since NW=2, the simulator issues two instructions at most each time, so it takes more cycles.

2) When NF=2, NW=4, NR=16, NB=4, total cycle is 39,

Since the decode queue is infinite and NW=4, the simulator will be hungry, which costs more cycles.

3) When NF=4, NW=4, NR=4, NB=4, total cycle is 48,

When NF=4, NW=4, NR=8, NB=4, total cycle is 38,

When NF=4, NW=4, NR=32, NB=4, total cycle is 38,

The conclusion is that when NR is greater than 8, it means that ROB will not be full during operation, so when NR is greater than 8, its effect is the same as NR=32. When it is equal to 4, it means that the ROB will be full during operation, so more cycles of stall are needed.