Theory Assignment

Tomasulo Implementation Analysis for functional blocks

MATHEMATICAL PROBLEM : FACTORIAL

Submitted by:
CED17I038 - Sreepathy Jayanand

CONTENT

INTRODUCTION:

Tomasulo's algorithm is a computer architecture hardware algorithm for dynamic scheduling of instructions that allows in-order issue, out-of-order execution and out-of-order commit,and enables more efficient use of multiple execution units.

Tomasulo algorithm uses:

- **Register renaming** to correctly perform out of order execution. A register holds either the real value or the place holders value.If a real value is unavailable to a destination register during the issue stage, a placeholder value is initially used. The placeholder value is a tag indicating from which reorder buffer the value needed will be obtained. When the unit finishes and broadcasts the result on the CDB, the placeholder will be replaced with the real value.
- **Reservation stations** to control when an instruction can execute and to hold the information needed to execute a single instruction, including the operation and the operands.
- **Common data bus** to connect reservation stations directly to the functional units. This lets multiple units that are waiting for the result to proceed parallely. It also distributes the hazard detection and control execution.
- **Reorder buffer** to track the state of all inflight instructions in the pipeline. The role of the ROB is to provide the illusion to the programmer that his program executes in-order. After instructions are decoded and renamed they are then dispatched to the ROB and the Issue Queue and marked as busy.
- **Operand forwarding** is used to limit performance deficits which occur due to pipeline stalls. A data hazard can lead to a pipeline stall when the current operation has to wait for the results of an earlier operation which has not yet finished. When a given instruction is finished we forward the operand to the restoration stations that need them

OVERVIEW OF TOMASULO IMPLEMENTATION:

Reorder Buffer:

ROB has fields which indicate whether the current entry is busy, which instruction is present, which state it is in, where the output has to be stored, what is the value to be stored, and the timing table index.

Reservation Stations:

Reservation station gives value if it is available(vj,vk) else it will tell us the location from where the values will be obtained(qj,qk) and busy will be set to no if both the values are available else it is yes.

Timing table:

Timing table gives the clock cycle at which an instruction is issued, started execution/memory, finished execution/memory, started write back, started commit and finished commit.

Register Alias Table(RAT):

Gives information from which register/ROB the required value has to be taken from.

Architectural registers(ARF):

The physical registers where the actual values reside.

Memory:

Gives the contents in the memory. The size is 256B(64W).

Load Store Queue:

Gives the type of instruction(LOAD/STORE), the rob destination, the address to where the value has to be stored or the address where the value has to be loaded, the constant which has to be added to the address field, the value that has to be loaded/ which has been brought back from the memory.

Mathematical problem: Finding factorial of a number.

Factorial of a positive integer n ,denoted by n!, is the product of all positive numbers less than n or equal to n. It is calculated using the equation:

n! = nx(n-1)x(n-2)x(n-3)x …….3x2x1

<u>Real world application of factorial:</u>

- **Factorial** function is used to count the number of ways we can choose things from a collection of things.
- It used to count the number of ways of arranging n distinct numbers.
- It is Used extensively in combinatorics.
- It is used in the computation of the irrational number ‘e’.
- It is also used in binomial theorem.

DEFAULT CODE SPECIFICATIONS:

In case if the user does not provide the specification, the tomasulo implementation works based on the default specification.

Integer adder =>Number of reservation stations is 2
Number of cycles required in execution is 1
Number of adder functional units is 2

Integer multiplier => Number of reservation stations is 2
Number of cycles required in execution is 1
Number of integer multiplier functional units is 2

Floating point adder =>Number of reservation stations is 3
Number of cycles required in execution is 3
Number of floating-point adder functional unit is 1

Floating point multiplier =>Number of reservation stations is 2
Number of cycles required in execution is 20
Number of floating-point multiplier functional unit is 1

Load or store =>Number of reservation stations is 3
Number of cycles required in execution is 1
Number of cycles required in memory is 4
Number of load-store functional units is 1

Logic unit =>Number of reservation stations is 2
Number of cycles required in execution is 3
Number of load-store functional units is 2

Rob entries =>Maximum number of rob entries is 128

ARM CODE:

The following ARM code finds the factorial of the number present in register R5:

```
LDR R1, R5
SUB R2,R1, #1
BE End
Loop:
Mul R1, R1, R2
Sub R2 R2 #1
BNE loop
End:
STR R1,R4
```

We assume, initially R5 has the value 5 and R4 has the final result.

INPUT TO THE TOMASULO IMPLEMENTATION:

The code given as input to the tomasulo implementation is:

```
int_adder 2 1 1
int_multiplier 2 1 1
fp_adder 3 3 1
fp_multiplier 2 20 1
load_store_unit 3 1 4 1
rob_entries 10
MEM 4 5
MEM 8 1
MEM 12 1
MEM 16 0
LD R1 1(R0)          #instruction 1
LD R2 2(R0)          #instruction 2
LD R3 3(R0)          #instruction 3
LD R4 4(R0)          #instruction 4
SUB R2 R1 R2         #instruction 5
BEQ R2 R4 3          #instruction 6
MULT R1 R1 R2        #instruction 7
SUB R2 R2 R3         #instruction 8
BNE R2 R3 -3         #instruction 9
SD R1 5(R0)          #instruction 10
```

ALGORITHM:

Step 1: Load all the values from the main memory to the registers.Let the number whose factorial has to be calculated be n ,and the register storing n be R1.

Step 2: Subtract 1 from the number whose factorial has to be calculated and store it in register R2.

Step 3: If the stored value in R2 is 0 go to step 7(when n is 1, n!=1), otherwise go to step 4.

Step 4: Multiply the values in register R1 and R2 and store it in R1.

Step 5: Subtract 1 from the value in register R2 and store it in R2.

Step 6: If the value in register R2 is 1 goto step 7,otherwise goto step 4.

Step 7: Store the result in register R1 to memory.

ANALYSIS OF THE INPUT CODE:

CODE SPECIFICATIONS:

The user can provide the specifications such as number of reservation stations, cycles required in execution, cycles required in memory, number of functional units for each functional unit in the input file in the following format:

int_adder 2 1 1 => The properties of integer adder
There are 2 reservation stations
Number of cycles required in execution is 1
Number of adder functional units is 2

int_multiplier 2 1 1 => The properties of integer multiplier
There are 2 reservation stations
Number of cycles required in execution is 1
Number of integer multiplier functional units is 2

fp_adder 3 3 1 => The properties of floating point adder
There are 3 reservation stations
Number of cycles required in execution is 3
Number of floating-point adder functional units is 1

fp_multiplier 2 20 1 => The properties of floating-point multiplier
There are 2 reservation stations
Number of cycles required in execution is 20
Number of floating-point multiplier functional units is 1

load_store_unit 3 1 4 1 =>The properties of load store unit
There are 3 reservation stations

Number of cycles required in execution is 1
Number of cycles required in memory is 4
Number of load-store functional units is 1

rob_entries 10 => Maximum instruction in the rob is 10.

INITIALIZING THE MEMORY:

MEM 4 5 => Instruction stores value 5 at address 0x4

MEM 8 1 => Instruction stores value 1 at address 0x8

MEM 12 1 => Instruction stores value 1 at address 0xc

MEM 16 0 => Instruction stores value 0 at address 0x10

INSTRUCTIONS AND THEIR MEANING:

LD R1 1(R0) => Instruction to load value at address 0x4 in register R1
I.e value 5 in R1 (address is calculated as 0+1*4=4=0x4 i.e by extracting the digit with R which is 0 and adding the product of 4 and the constant 1).

LD R2 2(R0) => Instruction to load value at address 0x8 in register R2
i.e value 1 in R2 (address is calculated as 0+2*4=8=0x8 i.e by extracting the digit with R which is 0 and adding the product of 4 and the constant 2).

LD R3 3(R0) => Instruction to load value at address 0xc in register R2
i.e value 1 in R3 (address is calculated as

0+3*4=8=0xc i.e by extracting the digit with R which is 0 and adding the product of 4 and the constant 3).

LD R4 4(R0) => Instruction to load value at address 0x10 in register R2 i.e value 0 in R4 (address is calculated as 0+4*4=16=0x10 i.e by extracting the digit with R which is 0 and adding the product of 4 and the constant 4).

SUB R2 R1 R2 => Instruction to subtract R2 from R1 and stores it in R2

BEQ R2 R4 3 => If R2 and R4 is equal then jump to the instruction SD R1 5(R0)

MULT R1 R1 R2 => Instruction to multiply R1 and R2 and stores it in R1

SUB R2 R2 R3 => instruction to subtract R3 from R2 and stores it in R2

BNE R2 R3 -3 => If R2 and R3 are not equal then jump to the instruction MULT R1 R1 R2

SD R1 5(R0) => Instruction to store the value in R1 to the memory address 0x14 (address is calculated as 0+4*5=20=0x14 i.e by extracting the digit with R which is 0 and adding the product of 4 and the constant 5).

HAZARD ANALYSIS:

Hazard is the situation that prevents the next instruction in the instruction stream from executing during its designated clock cycle. Hazards reduce the performance from the ideal speedup gained by pipelining.

1. Data hazard:

Data hazards occur when instructions that exhibit data dependence modify data in different stages of a pipeline. Different types of data hazards are WAW, WAR, and RAW hazards.

Tomasulo's algorithm eliminates the WAW and WAR hazards by renaming the instructions in the issue stage. It eliminates RAW hazards by delaying the instructions in the execute stage until all of their operands are available

2. Structural hazard:

A structural hazard occurs when two (or more) instructions that are already in the pipeline need the same resource. The result is that instruction must be executed in series rather than parallel for a portion of pipeline. Structural hazards are sometimes referred to as resource hazards.

3. Control hazard:

Control hazard occurs when the pipeline makes wrong decisions on branch prediction and therefore brings instructions into the pipeline that must subsequently be discarded. The term branch hazard also refers to a control hazard.

The hazards that can occur in the above code are:

- ***Data Hazard:***
  - WAW Hazard:
    1. wrt R1:

       LD R1 1(R0) #instruction 1

       MULT R1 R1 R2 #instruction 7
    2. wrt R1:

LD R1 1(R0) #instruction 1
SD R1 5(R0) #instruction 10

3. wrt R1:

MULT R1 R1 R2 #instruction 7
SD R1 5(R0) #instruction 10

4. wrt R2:

LD R2 2(R0) #instruction 2
SUB R2 R1 R2 #instruction 5

➢ WAR Hazard:

1. wrt R1

SUB R2 R1 R2 #instruction 5
MULT R1 R1 R2 #instruction 7

2. wrt R2

SUB R2 R1 R2 #instruction 5
SUB R2 R2 R3 #instruction 8

➢ RAW Hazard:

1. wrt R2

SUB R2 R1 R2 #instruction 5
BEQ R2 R4 3 #instruction 6

2.wrt R2

SUB R2 R2 R3 #instruction 8
BNE R2 R3 -3 #instruction 9

3. wrt R1

MULT R1 R1 R2 #instruction 7
SD R1 5(R0) #instruction 10

4.wrt R2

LD R2 2(R0) #instruction 2
BEQ R2 R4 3 #instruction 6

5.wrt R4

LD R4 4(R0) #instruction 2

BEQ R2 R4 3 #instruction 6

- ***Structural hazard:***

1.Only 1 functional unit for load instruction
When instruction 1 executes instruction 2 stalls

```
LD R1 1(R0)     #instruction 1
LD R2 2(R0)     #instruction 2
```

2.Only 1 functional unit for load instruction
When instruction 2 executes instruction 2 stalls

```
LD R2 2(R0)     #instruction 2
LD R3 3(R0)     #instruction 3
```

3.Only 3 reservation station for load instruction
When instructions 1,2,3 are present in reservation stations instruction 4 stalls

```
LD R1 1(R0)     #instruction 1
LD R2 2(R0)     #instruction 2
LD R3 3(R0)     #instruction 3
LD R4 4(R0)     #instruction 4
```

4.Only 1 functional unit for load instruction
When instruction 3 executes instruction 4 stalls

```
LD R3 3(R0)     #instruction 3
LD R4 4(R0)     #instruction 4
```

5.Only 1 instruction can access memory at a time
When instruction 1 accesses memory instruction 2 stalls

```
LD R1 1(R0)     #instruction 1
LD R2 2(R0)     #instruction 2
```

6.Only 1 instruction can access memory at a time
When instruction 2 accesses memory instruction 3 stalls

```
LD R2 2(R0)     #instruction 2
LD R3 3(R0)     #instruction 3
```

7.Only 1 instruction can access memory at a time
When instruction 3 accesses memory instruction 4 stalls

```
LD R3 3(R0)     #instruction 3
```

LD R4 4(R0) #instruction 4

❖ ***Control Hazard:***

The following two branch instructions can lead to control hazard :

1.BEQ R2 R4 3 #instruction 6

2.BNE R2 R3 -3 #instruction 9

ANALYSIS OF STALL:

Instructions and the number of cc it takes for the execution :

In Tomasulo algorithm we follow the policy of inorder issue,out-of-order execution and in-order commit since ROB is implemented. An instruction moves through 5 stages. They are issue, execution, memory, writeback and commit.

Following are the terms used in the analysis:

ISSUE : Clock at which an instruction is issued.

EX_S : Clock at which an instruction starts execution.

EX_F : Clock at which an instruction finishes execution.

MEM_S : Clock at which an instruction starts accessing memory.

MEM_F : Clock at which an instruction finishes accessing memory.

WB : Clock at which an instruction result is ready to be written back to the register or memory.

COMMIT_S : Clock at which an instruction starts writing its result back to register.

COMMIT_F : Clock at which an instruction finishes writing its result back to register.

Expected : Clock at which an instruction is expected to move to issue, execute,memory,writeback,commit stages.

Actual : Clock at which an instruction actually moves to issue, execute, memory, writeback, commit stages in the tomasulo's implementation.

To resolve a hazard, execution of an instruction must be delayed and this delay is called stall. Stalls also occur due to unavailability of resources like functional units, reservation stations, etc.

Assuming the value whose factorial is to be found is 5.

The analysis of stall for every instruction is as follows:

LD R1 1(R0) #instruction 1

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
expected	**1**	**2**	**3**	**4**	**7**	**8**	**9**	**9**
Actual	**1**	**2**	**3**	**4**	**7**	**8**	**9**	**9**

No stall as the expected clock cycles and actual clock cycles are the same.

LD R2 2(R0) #instruction 2

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
expected	**2**	**3**	**4**	**5**	**8**	**9**	**10**	**10**
Actual	**2**	**4**	**5**	**8**	**11**	**12**	**13**	**13**

Stall as the actual clock cycles > expected clock cycles
There is a delay of 1 (i.e 3rd cc) clock cycle between the ISSUE and EX_S, and a delay of 2 (i.e 6th cc and 7th cc) clock cycle between the EX_F and MEM_S.

Stall is due to unavailability of the load store functional unit as there is only one functional unit for load store unit. Stall also occur in the memory stage since at a time only one instruction can access the memory

LD R3 3(R0) #instruction 3

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
expected	**3**	**4**	**5**	**6**	**9**	**10**	**11**	**11**
Actual	**3**	**6**	**7**	**12**	**15**	**16**	**17**	**17**

Stall as the actual clock cycles > expected clock cycles
There is a delay of 2 (i.e 4th cc and 5th cc) clock cycle between the ISSUE and EX_S, and a delay of 4 (i.e 8th,9th,10th and 11th cc)clock cycle between the EX_F and MEM_S.

Stall is due to unavailability of the load store functional unit as there is only one functional unit for load store unit. Stall also occur in the memory stage, since at a time only one instruction can access the memory

LD R4 4(R0) #instruction 4

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**4**	**5**	**6**	**7**	**10**	**11**	**12**	**12**
Actual	**9**	**10**	**11**	**16**	**19**	**20**	**21**	**21**

Stall as the actual clock cycles > expected clock cycles
There is a delay of 5 (4th,5th,6th,7th,8th) clock cycles in the issue stage of the actual as compared to the expected .

Stall is due to unavailability of the reservation station for load-store unit. As there are only 3 reservation stations for load-store. So this instruction is stalled without issuing till a reservation station is free.

SUB R2 R1 R2 #instruction 5

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**10**	**11**	**11**	**-**	**-**	**12**	**13**	**13**
Actual	**10**	**13**	**13**	**-**	**-**	**14**	**21**	**21**

Stall as the actual clock cycles > expected clock cycles

There is a stall of 1 (12th cc) clock cycle between the ISSUE and the EX_S stage, and a stall of 6 clock cycles between WB and the COMMIT_S stage.

Stall is due to the data dependency with the instruction 2 and instruction 6. This is to avoid data hazard(RAW and WAW hazards).

BEQ R2 R4 3 #instruction 6

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**11**	**12**	**12**	**-**	**-**	**13**	**14**	**14**
Actual	**11**	**21**	**21**	**-**	**-**	**22**	**23**	**23**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 10 clock cycles between the ISSUE and the EX_S stage.

Stall is due to the data dependency with the instruction 4. This is to avoid data hazard(RAW and WAW hazards).

The branch condition turns false resulting in the execution of the 7th instruction.

MULT R1 R1 R2 #instruction 7

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**12**	**13**	**13**	**-**	**-**	**14**	**15**	**15**
Actual	**22**	**23**	**23**	**-**	**-**	**24**	**25**	**25**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 9 clock cycles in the issue stage of the actual as compared to the expected .

Stall is to avoid the control hazard caused due to instruction 6.

SUB R2 R2 R3 #instruction 8

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**23**	**24**	**24**	**-**	**-**	**25**	**26**	**26**
Actual	**23**	**24**	**24**	**-**	**-**	**25**	**26**	**26**

No Stall as the actual clock cycles = expected clock cycles

BNE R2 R3 -3 #instruction 9

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**24**	**25**	**25**	**-**	**-**	**26**	**26**	**26**
Actual	**24**	**26**	**26**	**-**	**-**	**27**	**27**	**27**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 1 clock cycle between the ISSUE and EX_S stage.

Stall is to avoid data hazard(RAW hazard) since this instruction is dependent on instruction 8.

The branch condition turns true resulting in the execution of the 7th instruction again.

MULT R1 R1 R2 #instruction 7

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**25**	**26**	**26**	**-**	**-**	**27**	**28**	**28**
Actual	**27**	**28**	**28**	**-**	**-**	**29**	**30**	**30**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 1 clock cycle in the issue stage of the actual as compared to that of expected.
Stall is to avoid the control hazard caused due to instruction 9.

SUB R2 R2 R3 #instruction 8

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**28**	**29**	**29**	**-**	**-**	**30**	**31**	**31**
Actual	**28**	**29**	**29**	**-**	**-**	**30**	**31**	**31**

No Stall as the actual clock cycle values = expected clock cycle values

BNE R2 R3 -3 #instruction 9

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**29**	**30**	**30**	**-**	**-**	**31**	**31**	**31**
Actual	**29**	**31**	**31**	**-**	**-**	**32**	**32**	**32**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 1 clock cycle between the ISSUE and EX_S stage.

Stall is to avoid data hazard(RAW hazard) since this instruction is dependent on instruction 8.

The branch condition turns true resulting in the execution of the 7th instruction again.

MULT R1 R1 R2 #instruction 7

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F

Expected	**30**	**31**	**31**	**-**	**-**	**32**	**33**	**33**
Actual	**32**	**33**	**33**	**-**	**-**	**34**	**35**	**35**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 1 clock cycle in the issue stage of the actual as compared to that of expected.

Stall is to avoid the control hazard caused due to instruction 9.

SUB R2 R2 R3 #instruction 8

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**33**	**34**	**34**	**-**	**-**	**35**	**36**	**36**
Actual	**33**	**34**	**34**	**-**	**-**	**35**	**36**	**36**

No Stall as the actual clock cycle values = expected clock cycle values

BNE R2 R3 -3 #instruction 9

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**34**	**35**	**35**	**-**	**-**	**36**	**36**	**36**
Actual	**34**	**36**	**36**	**-**	**-**	**37**	**37**	**37**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 1 clock cycle between the ISSUE and EX_S stage.

Stall is to avoid data hazard(RAW hazard) since this instruction is dependent on instruction 8.

The branch condition turns false resulting in the execution of the 10th instruction.
SD R1 5(R0) #instruction 10

	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
Expected	**35**	**36**	**37**	**-**	**-**	**38**	**39**	**40**
Actual	**37**	**38**	**39**	**-**	**-**	**40**	**41**	**44**

Stall as the actual clock cycles > expected clock cycles
There is a stall of 1 clock cycle in the issue stage of the actual as compared to that of expected.

Stall is to avoid the control hazard caused due to instruction 9.

The COMMIT_F of the final instruction is at clock cycle 44 i.e it takes 44cc to complete the tomasulo implementation.

OUTPUT OF THE TOMASULO IMPLEMENTATION:

Specifications:

INTEGER ADDER PROPERTIES

Number of reservation stations: 2
Number of cycles in execution stage: 1
Number of function Units: 1

INTEGER MULTIPLIER PROPERTIES

Number of reservation stations: 2
Number of cycles in execution stage: 1
Number of function Units: 1

FP ADDER PROPERTIES

Number of reservation stations: 3
Number of cycles in execution stage: 3
Number of function Units: 1

FP MULTIPLIER PROPERTIES

Number of reservation stations: 2
Number of cycles in execution stage: 20
Number of function Units: 1

LOAD STORE UNIT PROPERTIES

Number of reservation stations: 3
Number of cycles in execution stage: 2
Number of cycles in memory stage: 4
Number of function Units: 1

ROB PROPERTIES

Number of rob entries: 10

Clock cycle:0

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

FLOATING POINT ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

FLOATING POINT MULTIPLIER RS

BUSY OP DEST Vj Vk Qj Qk
- - - - - - -

LOGIC UNIT RS

BUSY OP DEST Vj Vk Qj Qk
- - - - - - -

LOAD STORE QUEUE

DEST TYPE VSD QSD VAddr QAddr CONST ADDR VAL FWD
- - - - - - - - - -

TIMING TABLE

PC INSTRUCTION ISSUE EX_S EX_F MEM_S MEM_F WB COMMIT_S COMMIT_F
- - - - - - - - - -

MEMORY VALUES

Initial Memory Contents
0x00004 -> 5
0x00008 -> 1
0x0000c -> 1
0x00010 -> 0

INTEGER ARF

INT ARF R0-R31 -> 0

FLOATING POINT ARF

FLOAT ARF F0-F31 -> 0

INTEGER RAT

R0 - R31 -> R0 - R31

FLOATING POINT RAT

F0 - F31 -> F0 - F31

Clock cycle:1

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	LD R1 1(R0)	ISSUE	R1	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB0	LD	-	-	0	-	1	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	-	-	-	-	-	-	-

INTEGER ARF

INT ARF R0-R31 -> 0

INTEGER RAT

R1 -> ROB0

Clock cycle:2

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	LD R1 1(R0)	EX	R1	-
ROB1	yes	LD R2 2(R0)	ISSUE	R2	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB0	LD	-	-	0	-	1	-	-	-

ROB1 LD - - 0 - 2 - - -

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	-	-	-	-	-
4	LD R2 2(R0)	2	-	-	-	-	-	-	-

INTEGER ARF

INT ARF R0-R31 -> 0

INTEGER RAT

R1 -> ROB0, R2 -> ROB1

Clock cycle:3

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	LD R1 1(R0)	EX	R1	-
ROB1	yes	LD R2 2(R0)	ISSUE	R2	-
ROB2	yes	LD R3 3(R0)	ISSUE	R3	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB0	LD	-	-	0	-	1	4	-	-
ROB1	LD	-	-	0	-	2	-	-	-
ROB2	LD	-	-	0	-	3	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	-	-	-	-	-
4	LD R2 2(R0)	2	-	-	-	-	-	-	-
8	LD R3 3(R0)	3	-	-	-	-	-	-	-

INTEGER ARF

INT ARF R0-R31 -> 0

INTEGER RAT

R1 -> ROB0, R2 -> ROB1, R3 -> ROB2

Clock cycle:4

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	LD R1 1(R0)	MEM	R1	-
ROB1	yes	LD R2 2(R0)	EX	R2	-
ROB2	yes	LD R3 3(R0)	ISSUE	R3	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB0	LD	-	-	0	-	1	4	-	-
ROB1	LD	-	-	0	-	2	-	-	-
ROB2	LD	-	-	0	-	3	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	-	-	-
4	LD R2 2(R0)	2	4	5	-	-	-	-	-
8	LD R3 3(R0)	3	-	-	-	-	-	-	-

INTEGER ARF

R0 - R31 -> 0

INTEGER RAT

R1 -> ROB0, R2 -> ROB1, R3 -> ROB2

Clock cycle:5

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	LD R1 1(R0)	MEM	R1	-
ROB1	yes	LD R2 2(R0)	EX	R2	-
ROB2	yes	LD R3 3(R0)	ISSUE	R3	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB0	LD	-	-	0	-	1	4	-	-
ROB1	LD	-	-	0	-	2	-	-	-

ROB2 LD - - 0 - 3 - - -

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	-	-	-
4	LD R2 2(R0)	2	4	5	-	-	-	-	-
8	LD R3 3(R0)	3	-	-	-	-	-	-	-

INTEGER ARF

R0 - R31 -> 0

Clock cycle:6

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	LD R1 1(R0)	MEM	R1	-
ROB1	yes	LD R2 2(R0)	EX	R2	-
ROB2	yes	LD R3 3(R0)	EX	R3	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB0	LD	-	-	0	-	1	4	-	-
ROB1	LD	-	-	0	-	2	8	-	-
ROB2	LD	-	-	0	-	3	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	-	-	-
4	LD R2 2(R0)	2	4	5	-	-	-	-	-
8	LD R3 3(R0)	3	6	7	-	-	-	-	-

INTEGER ARF

R0 - R31 -> 0

INTEGER RAT

R1 -> ROB0, R2 -> ROB1, R3 -> ROB2

Clock cycle:7

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	LD R1 1(R0)	MEM	R1	-
ROB1	yes	LD R2 2(R0)	EX	R2	-
ROB2	yes	LD R3 3(R0)	EX	R3	-
ROB3	no	-	-	-	-

ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB0	LD	-	-	0	-	1	4	-	-
ROB1	LD	-	-	0	-	2	8	-	-
ROB2	LD	-	-	0	-	3	12	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	-	-	-
4	LD R2 2(R0)	2	4	5	-	-	-	-	-
8	LD R3 3(R0)	3	6	7	-	-	-	-	-

INTEGER ARF

R0 - R31 -> 0

INTEGER RAT

R1 -> ROB0, R2 -> ROB1, R3 -> ROB2

Clock cycle:8

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	LD R1 1(R0)	WB	R1	5.0
ROB1	yes	LD R2 2(R0)	MEM	R2	-
ROB2	yes	LD R3 3(R0)	EX	R3	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB1	LD	-	-	0	-	2	8	-	-
ROB2	LD	-	-	0	-	3	12	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	-	-
4	LD R2 2(R0)	2	4	5	8	11	-	-	-
8	LD R3 3(R0)	3	6	7	-	-	-	-	-

INTEGER ARF

R0 - R31 -> 0

INTEGER RAT

R1 -> ROB0, R2 -> ROB1, R3 -> ROB2

Clock cycle:9

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	yes	LD R2 2(R0)	MEM	R2	-
ROB2	yes	LD R3 3(R0)	EX	R3	-
ROB3	yes	LD R4 4(R0)	ISSUE	R4	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk

-	-	-	-	-	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB1	LD	-	-	0	-	2	8	-	-
ROB2	LD	-	-	0	-	3	12	-	-
ROB3	LD	-	-	0	-	4	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	-	-	-
8	LD R3 3(R0)	3	6	7	-	-	-	-	-
12	LD R4 4(R0)	9	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5

INTEGER RAT

R2 -> ROB1, R3 -> ROB2, R4 -> ROB3

Clock cycle:10

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	yes	LD R2 2(R0)	MEM	R2	-
ROB2	yes	LD R3 3(R0)	EX	R3	-
ROB3	yes	LD R4 4(R0)	EX	R4	-
ROB4	yes	SUB R2 R1 R2	ISSUE	R2	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	SUB	ROB4	5	-	-	ROB1

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
-	-	-	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB1	LD	-	-	0	-	2	8	-	-
ROB2	LD	-	-	0	-	3	12	-	-
ROB3	LD	-	-	0	-	4	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	-	-	-

8	LD R3 3(R0)	3	6	7	-	-	-	-	-
12	LD R4 4(R0)	9	10	11	-	-	-	-	-
16	SUB R2 R1 R2	10	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5

INTEGER RAT

R2 -> ROB4, R3 -> ROB2, R4 -> ROB3

Clock cycle:11

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	yes	LD R2 2(R0)	MEM	R2	-
ROB2	yes	LD R3 3(R0)	EX	R3	-
ROB3	yes	LD R4 4(R0)	EX	R4	-
ROB4	yes	SUB R2 R1 R2	ISSUE	R2	-
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	SUB	ROB4	5	-	-	ROB1
yes	BEQ	ROB5	-	-	ROB4	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB1	LD	-	-	0	-	2	8	-	-
ROB2	LD	-	-	0	-	3	12	-	-
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	-	-	-
8	LD R3 3(R0)	3	6	7	-	-	-	-	-
12	LD R4 4(R0)	9	10	11	-	-	-	-	-
16	SUB R2 R1 R2	10	-	-	-	-	-	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5

INTEGER RAT

R2 -> ROB4, R3 -> ROB2, R4 -> ROB3

Clock cycle:12

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	LD R2 2(R0)	WB	R2	1.0
ROB2	yes	LD R3 3(R0)	MEM	R3	-
ROB3	yes	LD R4 4(R0)	EX	R4	-
ROB4	yes	SUB R2 R1 R2	ISSUE	R2	-
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-

ROB9 no - - - -

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB4	5	1.0	-	-
yes	BEQ	ROB5	-	-	ROB4	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB2	LD	-	-	0	-	3	12	-	-
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	-	-
8	LD R3 3(R0)	3	6	7	12	15	-	-	-
12	LD R4 4(R0)	9	10	11	-	-	-	-	-
16	SUB R2 R1 R2	10	-	-	-	-	-	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5

INTEGER RAT

R2 -> ROB4, R3 -> ROB2, R4 -> ROB3

Clock cycle:13

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	yes	LD R3 3(R0)	MEM	R3	-
ROB3	yes	LD R4 4(R0)	EX	R4	-
ROB4	yes	SUB R2 R1 R2	EX	R2	-
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB4	5	1.0	-	-
yes	BEQ	ROB5	-	-	ROB4	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB2	LD	-	-	0	-	3	12	-	-
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	-	-	-
12	LD R4 4(R0)	9	10	11	-	-	-	-	-
16	SUB R2 R1 R2	10	13	13	-	-	-	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1

INTEGER RAT

R2 -> ROB4, R3 -> ROB2, R4 -> ROB3

Clock cycle:14

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	yes	LD R3 3(R0)	MEM	R3	-
ROB3	yes	LD R4 4(R0)	EX	R4	-
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	BEQ	ROB5	4.0	-	-	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB2	LD	-	-	0	-	3	12	-	-
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	-	-	-
12	LD R4 4(R0)	9	10	11	-	-	-	-	-
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1

INTEGER RAT

R2 -> ROB4, R3 -> ROB2, R4 -> ROB3

Clock cycle:15

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	yes	LD R3 3(R0)	MEM	R3	-
ROB3	yes	LD R4 4(R0)	EX	R4	-
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	BEQ	ROB5	4.0	-	-	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB2	LD	-	-	0	-	3	12	-	-
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	-	-	-
12	LD R4 4(R0)	9	10	11	-	-	-	-	-
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1

INTEGER RAT

R2 -> ROB4, R3 -> ROB2, R4 -> ROB3

Clock cycle:16

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	LD R3 3(R0)	WB	R3	1.0
ROB3	yes	LD R4 4(R0)	MEM	R4	-
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-

ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	BEQ	ROB5	4.0	-	-	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	-	-
12	LD R4 4(R0)	9	10	11	16	19	-	-	-
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1

INTEGER RAT

R2 -> ROB4, R3 -> ROB2, R4 -> ROB3

Clock cycle:17

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	yes	LD R4 4(R0)	MEM	R4	-
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	BEQ	ROB5	4.0	-	-	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	-	-	-
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1

INTEGER RAT

R2 -> ROB4, R4 -> ROB3

Clock cycle:18

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	yes	LD R4 4(R0)	MEM	R4	-
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	BEQ	ROB5	4.0	-	-	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB3	LD	-	-	0	-	4	16	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17

12	LD R4 4(R0)	9	10	11	16	19	-	-	-
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1, R3->1

INTEGER RAT

R2 -> ROB4, R4 -> ROB3

Clock cycle:19

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	yes	LD R4 4(R0)	MEM	R4	-
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	BEQ	ROB5	4.0	-	-	ROB3

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD

ROB3 LD - - 0 - 4 16 - -

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	-	-	-
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1, R3->1

INTEGER RAT

R2 -> ROB4, R4 -> ROB3

Clock cycle:20

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	LD R4 4(R0)	WB	R4	0.0
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	ISSUE	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	BEQ	ROB5	4.0	0.0	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	-	-
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1, R3->1

INTEGER RAT

R2 -> ROB4, R4 -> ROB3

Clock cycle:21

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	SUB R2 R1 R2	WB	R2	4.0
ROB5	yes	BEQ R2 R4 3	EX	36	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-

ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	BEQ	ROB5	4.0	0.0	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	-	-
20	BEQ R2 R4 3	11	21	21	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 1, R3->1

INTEGER RAT

R2 -> ROB4

Clock cycle:22

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-

ROB4	no	-	-	-	-
ROB5	yes	BEQ R2 R4 3	EX	36	-
ROB6	yes	MULT R1 R1 R2	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	-	-	-
24	MULT R1 R1 R2	22	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 4, R3->1,R4->0

INTEGER RAT

R1 -> ROB6

Clock cycle:23

ROB

BUSY	INSTRUCTION	STATE	DESTINATION	VALUE

ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	yes	MULT R1 R1 R2	EX	R1	-
ROB7	yes	SUB R2 R2 R3	ISSUE	R2	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB7	4	1	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	-	-	-
28	SUB R2 R2 R3	23	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 4, R3->1, R4->0

INTEGER RAT

R1 -> ROB6, R2 -> ROB7

Clock cycle:24

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	yes	MULT R1 R1 R2	EX	R1	20
ROB7	yes	SUB R2 R2 R3	ISSUE	R2	-
ROB8	yes	BNE R2 R3 -1	ISSUE	24	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB7	4	1	-	-
yes	BNE	ROB8	-	1	ROB7	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	-	-
28	SUB R2 R2 R3	23	24	24	-	-	-	-	-
32	BNE R2 R3 -3	24	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 5, R2 -> 4, R3->1, R4->0

INTEGER RAT

R1 -> ROB6, R2 -> ROB7

Clock cycle:25

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	yes	SUB R2 R2 R3	ISSUE	R2	3
ROB8	yes	BNE R2 R3 -1	ISSUE	24	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	BNE	ROB8	3	1	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21

16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	-	-
28	SUB R2 R2 R3	23	24	24	-	-	25	-	-
32	BNE R2 R3 -3	24	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 20, R2 -> 4, R3->1, R4->0

INTEGER RAT

R2 -> ROB7

Clock cycle:26

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	yes	BNE R2 R3 -1	EX	24	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	BNE	ROB8	3	1	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	-	-	-

INTEGER ARF

R1 -> 20, R2 -> 3, R3->1, R4->0

INTEGER RAT

R0 - R31 -> R0 - R31

Clock cycle:27

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	yes	MULT R1 R1 R2	ISSUE	R1	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
No	MULT	ROB9	20	3	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 20, R2 -> 3, R3->1, R4->0

INTEGER RAT

R1 -> ROB9

Clock cycle:28

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	SUB R2 R2 R3	ISSUE	R2	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	yes	MULT R1 R1 R2	EX	R1	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB0	3	1	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
yes	MULT	ROB9	20	3	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	-	-	-
28	SUB R2 R2 R3	28	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 20, R2 -> 3, R3->1, R4->0

INTEGER RAT

R1 -> ROB9, R2 -> ROB0

Clock cycle:29

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	yes	SUB R2 R2 R3	EX	R2	-
ROB1	yes	BNE R2 R3 -3	ISSUE	24	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	MULT R1 R1 R2	WB	R1	60

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB0	3	1	-	-
yes	BNE	ROB1	-	1	ROB0	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	-	-
28	SUB R2 R2 R3	28	29	29	-	-	-	-	-
32	BNE R2 R3 -3	29	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 20, R2 -> 3, R3->1, R4->0

INTEGER RAT

R1 -> ROB9, R2 -> ROB0

Clock cycle:30

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	SUB R2 R2 R3	WB	R2	-
ROB1	yes	BNE R2 R3 -3	ISSUE	24	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-

ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	BNE	ROB1	2	1	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 60, R2 -> 3, R3->1, R4->0

INTEGER RAT

R1 -> ROB9, R2 -> ROB0

Clock cycle:31

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	yes	BNE R2 R3 -3	ISSUE	24	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	BNE	ROB1	2	1	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26

32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	-	-	-

INTEGER ARF

R1 -> 60, R2 -> 2, R3->1, R4->0

INTEGER RAT

R1 - R31 -> R1 - R31

Clock cycle:32

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	yes	MULT R1 R1 R2	ISSUE	R1	
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	MULT	ROB2	60	2	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32

INTEGER ARF

R1 -> 60, R2 -> 2, R3->1, R4->0

INTEGER RAT

R1 -> ROB2

Clock cycle:33

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	yes	MULT R1 R1 R2	EX	R1	-
ROB3	yes	SUB R2 R2 R3	ISSUE	R2	-
ROB4	no	-	-	-	-

ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB3	2	1	-	-

INTEGER MULTIPLIER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	MULT	ROB2	60	2	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	-	-	-
28	SUB R2 R2 R3	33	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 60, R2 -> 2, R3->1, R4->0

INTEGER RAT

R1 -> ROB2, R2 -> ROB3

Clock cycle:34

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	MULT R1 R1 R2	WB	R1	120
ROB3	yes	SUB R2 R2 R3	EX	R2	-
ROB4	yes	BNE R2 R3 -3	ISSUE	24	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	SUB	ROB3	2	1	-	-
yes	BNE	ROB4	-	1	ROB3	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23

24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	-	-
28	SUB R2 R2 R3	33	34	34	-	-	-	-	-
32	BNE R2 R3 -3	34	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 60, R2 -> 2, R3->1, R4->0

INTEGER RAT

R1 -> ROB2, R2 -> ROB3

Clock cycle:35

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	SUB R2 R2 R3	WB	R2	-
ROB4	yes	BNE R2 R3 -3	ISSUE	24	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY OP DEST Vj Vk Qj Qk

no BNE ROB4 1 1 - -

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 120, R2 -> 2, R3->1, R4->0

INTEGER RAT

R2 -> ROB3

Clock cycle:36

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-

ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	yes	BNE R2 R3 -3	EX	24	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

INTEGER ADDER RS

BUSY	OP	DEST	Vj	Vk	Qj	Qk
no	BNE	ROB4	1	1	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	36	36	-	-	-	-	-

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:37

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	yes	SD R1 5(R0)	ISSUE	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB5	SD	120	-	0	-	5	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36

32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	-	-	-	-	-	-	-

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:38

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	yes	SD R1 5(R0)	EX	20	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB5	SD	120	-	0	-	5	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22

20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	38	39	-	-	-	-	-

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:39

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	yes	SD R1 5(R0)	EX	20	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB5	SD	120	-	0	-	5	20	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	38	39	-	-	-	-	-

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:40

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	SD R1 5(R0)	WB	20	120
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB5	SD	120	-	0	-	5	20	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	38	39	-	-	40	-	-

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:41

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-

ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB5	SD	120	-	0	-	5	20	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	38	39	-	-	40	41	44

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:42

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB5	SD	120	-	0	-	5	20	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36

32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	38	39	-	-	40	41	44

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:43

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

LOAD STORE QUEUE

DEST	TYPE	VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD
ROB5	SD	120	-	0	-	5	20	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13
8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22

20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	38	39	-	-	40	41	44

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Clock cycle:44

ROB

	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE
ROB0	no	-	-	-	-
ROB1	no	-	-	-	-
ROB2	no	-	-	-	-
ROB3	no	-	-	-	-
ROB4	no	-	-	-	-
ROB5	no	-	-	-	-
ROB6	no	-	-	-	-
ROB7	no	-	-	-	-
ROB8	no	-	-	-	-
ROB9	no	-	-	-	-

TIMING TABLE

PC	INSTRUCTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R0)	1	2	3	4	7	8	9	9
4	LD R2 2(R0)	2	4	5	8	11	12	13	13

8	LD R3 3(R0)	3	6	7	12	15	16	17	17
12	LD R4 4(R0)	9	10	11	16	19	20	21	21
16	SUB R2 R1 R2	10	13	13	-	-	14	22	22
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23
24	MULT R1 R1 R2	22	23	23	-	-	24	25	25
28	SUB R2 R2 R3	23	24	24	-	-	25	26	26
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27
24	MULT R1 R1 R2	27	28	28	-	-	29	30	30
28	SUB R2 R2 R3	28	29	29	-	-	30	31	31
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32
24	MULT R1 R1 R2	32	33	33	-	-	34	35	35
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36
32	BNE R2 R3 -3	34	36	36	-	-	37	37	37
36	SD R1 5(R0)	37	38	39	-	-	40	41	44

MEMORY VALUES

Initial Memory Contents
0x00004 -> 5
0x00008 -> 1
0x0000c -> 1
0x00010 -> 0
0x00014->120.0

INTEGER ARF

R1 -> 120, R2 -> 1, R3->1, R4->0

Total execution time = 44 clock cycles.

RESULT :

Performed step by step analysis of computing factorial of a number using Tomasulo algorithm and performed analysis of theoretical execution parameters vs actual execution parameters.

Calculated 5! using the algorithm and took 44 clock cycles.