Theory Assignment

Tomasulo Implementation Analysis for functional blocks

MATHEMATICAL PROBLEM: FACTORIAL

Submitted by: CED17I038 - Sreepathy Jayanand

CONTENT

1. INTRODUCTION	2
2. OVERVIEW OF TOMASULO IMPLEMENTATION	3
3. MATHEMATICAL PROBLEM	4
4. DEFAULT CODE SPECIFICATION	5
5. ARM CODE	6
6. INPUT TO THE TOMASULO IMPLEMENTATION	7
7. ALGORITHM	8
8. ANALYSIS OF THE INPUT CODE	9
9. HAZARD ANALYSIS	12
10. ANALYSIS OF STALL	16
11. OUTPUT OF THE TOMASULO IMPLEMENTATION	24
12. RESULT	83

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:

$$n! = nx(n-1)x(n-2)x(n-3)x \dots 3x2x1$$

Real world application of factorial:

- 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
                      #instruction 3
LD R3 3(R0)
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
                      #instruction 9
BNE R2 R3 -3
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:

<u>int_adder 2 1 1</u>

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

<u>load_store_unit 3 1 4 1</u> =>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).

<u>LD R3 3(R0)</u> => 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) SD R1 5(R0)	#instruction 1 #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:	
> RAW Hazard: 1. wrt R2	
	#instruction 5
1. wrt R2	#instruction 5 #instruction 6
1. wrt R2 SUB R2 R1 R2	
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3	
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2	#instruction 6
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2 SUB R2 R2 R3	#instruction 6 #instruction 8
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2 SUB R2 R2 R3 BNE R2 R3 -3	#instruction 6 #instruction 8
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2 SUB R2 R2 R3 BNE R2 R3 -3 3. wrt R1	#instruction 6 #instruction 8 #instruction 9
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2 SUB R2 R2 R3 BNE R2 R3 -3 3. wrt R1 MULT R1 R1 R2	#instruction 6 #instruction 8 #instruction 9 #instruction 7
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2 SUB R2 R2 R3 BNE R2 R3 -3 3. wrt R1 MULT R1 R1 R2 SD R1 5(R0) 4.wrt R2 LD R2 2(R0)	#instruction 6 #instruction 8 #instruction 9 #instruction 7
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2 SUB R2 R2 R3 BNE R2 R3 -3 3. wrt R1 MULT R1 R1 R2 SD R1 5(R0) 4.wrt R2	#instruction 6 #instruction 8 #instruction 9 #instruction 7 #instruction 10
1. wrt R2 SUB R2 R1 R2 BEQ R2 R4 3 2.wrt R2 SUB R2 R2 R3 BNE R2 R3 -3 3. wrt R1 MULT R1 R1 R2 SD R1 5(R0) 4.wrt R2 LD R2 2(R0)	#instruction 6 #instruction 8 #instruction 9 #instruction 7 #instruction 10 #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.
 Clock at which an instruction finishes accessing memory.
 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 0 clock cycles in the issue stage of the actual 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	сомміт_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 Qi Qk **INTEGER MULTIPLIER RS** BUSY OP DEST Vj Vk Qi Qk FLOATING POINT ADDER RS BUSY OP DEST Vj Vk Qj Qk

FLOATING POINT MULTIPLIER RS

BUSY -	OP -	DEST -	Vj -	Vk -	Qj -	Qk -							
						UNIT RS							
BUSY -	OP -	DEST -	Vj										
						RE QUEU	 JE						
DEST	DEST TYPE VSD QSD VAddr QAddr CONST ADDR VAL FWD												
					TIMING	TABLE							
PC IN	ISTRUC1 -	ΓΙΟΝ ISS -	UE EX	K_S E -	EX_F M -	EM_S ME -	EM_F WB COMMIT_S COMMIT_F 						
				N	MEMORY	/ VALUES	} 						
Initial M 0x0000 0x0000 0x0000 0x0001	8 -> 1 c -> 1	ontents											
					INTEG	ER ARF							
INT AR	INT ARF R0-R31 -> 0												
	FLOATING POINT ARF												
FLOAT	ARF F0	-F31 -> 0											
					INTEG	ER RAT							

LD R1 1(R0) 1 -INTEGER ARF INT ARF R0-R31 -> 0

					INTEG	SER RAT		
R1 -> R0	DB0							
Clock	cycle:2							
					F	 ROB		
	BUSY	INST	RUCTIO	 NC	STATE	DESTINATIO	N VALUE	
ROB0	yes		R1 1(R0		EX	R1	-	
ROB1	yes	LD F	R2 2(R0))	ISSUE	R2	-	
ROB2	no		-		-	-	-	
ROB3	no		-		-	-	-	
ROB4	no		-		-	-	-	
ROB5	no		-		-	-	-	
ROB6	no		-		-	-	-	
ROB7	no		-		-	-	-	
ROB8	no		-		-	-	-	
ROB9	no 		-		-	-	- 	
				I	NTEGER	ADDER RS		
BUSY	OP [DEST	 Vj	 Vk	Qj	Qk		
-	-	-	-	-	-	-		
				INT	EGER M	ULTIPLIER R	S	
BUSY	OP [DEST	 Vj	 Vk	Qj	Qk		

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

ROB	31	LD -	-	0	-	2	-	-	-	
						NG TABI				
	INS ⁻ LD		ISSUE 1	EX_S 2	EX_F	MEM_S - -	MEM_F - -	WB - -	-	COMMIT_F - -
					INTE	EGER AR				
INT A	ARF I	R0-R31 -> 0								
					INTE	EGER RA				
R1 ->		B0, R2 -> R								
Clo	ck c	sycle:3								
						ROB				
ROB ROB ROB ROB ROB ROB ROB	31 32 33 34 35 36 37	BUSY yes yes no no no no no no	INSTRU LD R1 1 LD R2 2 LD R3 3	(R0) (R0) (R0)	ISSUE ISSUE	<u> </u>	R1 R2 R3 RR RR RR RR RR		UE	
BUS	Y	OP DI	 EST \ -	 /j Vł 	 (Qj -	Qk -				

				INTE	GER M	ULTIPLIE	R RS			
BUSY	ОР	DES	T Vj	Vk	Qj	Qk				
-	-	-	-	-	-	-				
				LC)AD ST(ORE QUE	UE			
DEST		VSD	QSD	VAddr	QAddr	CONST	ADDR	VAL	FWD	
ROB0 ROB1	LD		-	0	-	2	4 -	-	-	
ROB2	LD	-	- 	0	-	3 	-	- 	- 	
					TIMIN	G TABLE				
0	NSTRUCT LD R1 1(LD R2 2(LD R3 3(R0) R0)	1	2	3	-	1EM_F \ - - -	WB CC - - -	DMMIT_S - - -	COMMIT_I - - -
					INTEG	ER ARF				
INT AF	RF R0-R3	I -> 0								
					INTEG	ER RAT				
 R1 ->	ROB0, R2	-> ROE	 31, R3 ->	 ROB2						
Cloc	k cycle:	<u>4</u>								
					 F	 ROB				

	BUS		NSTRU		STATE		NATION	VAL	.UE	
ROB0	-	yes LD R1 1(R0)			MEM		R1			
ROB1	ye		LD R2 2		EX		R2	-		
ROB2	ye		LD R3 3(R0)		ISSUE		R3	-		
ROB3	nc		-		-		-	-		
ROB4	nc		-		-		-	-		
ROB5	nc		-		-		-	-		
ROB6	no		-		-		-	-		
ROB7	no		-		-		-	-		
ROB8	no		-		-		-	-		
ROB9	no)	-		-		-	-		
				I	NTEGE	R ADDEI	R RS			
BUSY	OP	DES	 T V	 'j Vk	 : Qj	Qk				
-	-	-	-		-	-				
				INT	EGER N	IULTIPL	IER RS			
BUSY	 OP	DES	 T V	 'j Vk	 : Qj	Qk				
-	-	-	-		-	-				
				L	 .OAD ST	ORE QU	JEUE			
DEST	TYPE	 VSD	QSD	 VAddı	· · QAdd	r CONS	 ST ADE	 DR V	AL FW	 D
ROB0	LD	-	-	0	-	1	4	_		
ROB1	LD	_	_	0	_	2	_	-	_	
ROB2	LD	-	-	0	-	3	-	-	-	
					TIMIN	IG TABL	 .E			
PC IN	 NSTRUCT	ION I	SSUE	EX_S	EX_F	MEM_S	MEM_F	 WB	COMMI	 Г_S COMMIT_F
	LD R1 1(F		1	2			7	-	-	
	LD R2 2(F		2	4	5	-	-	-	-	-
	LD R3 3(F		3	-	-	-	-	-	-	-
					INTE	GER AR	 F			
R0 - R3	 31 -> 0									

INTEGER RAT													
R1 -> R	R1 -> ROB0, R2 -> ROB1, R3 -> ROB2												
Clock	Clock cycle:5 												
	ROB												
	BU	SY INST	TRUCT	 ΓΙΟΝ	STATE	DESTINAT	ION '	VALUE					
ROB0	ye	es LD	R1 1(F	₹0)	MEM	R1		-					
ROB1	y€				EX	R2		-					
ROB2	y€		R3 3(F		ISSUE	R3		-					
ROB3	no)	-		-	-		-					
ROB4	no)	-		-	-		-					
ROB5	no	no -			-	-		-					
ROB6	no	no			-	-		-					
ROB7	no)	-		-	-		-					
ROB8	no)	-		-	-		-					
ROB9	no)	-		-	-		-					
				IN	ITEGER	ADDER R	 S						
BUSY	OP -	DEST -	Vj -	Vk -	Qj -	Qk -							
				INTE	 GER MI	ULTIPLIER	RS						
BUSY	 ОР	DEST		 Vk		Ok							
-	-	-	- -	- -	Qj -	Qk -							
LOAD STORE QUEUE													
DEST ROB0 ROB1	TYPE LD LD	VSD Q - -	(SD - -	VAddr 0 0	QAddr - -	CONST 1 2	ADDR 4 -	VAL - -	FWD - -				

ROB2	LD		0	-	3	-						
TIMING TABLE												
PC I	INSTRUCTION	ON ISSUE	E EX S	EX F	MEM S	MEM F	WB	COMMIT_S	COMMIT F			
0	LD R1 1(R0		_			7	-	-	-			
4	LD R2 2(R0		4	5	-	-	-	-	-			
8	LD R3 3(R0	0) 3	-	-	-	-	-	-	-			
				INTE	GER AR	 RF						
R0 - R	R31 -> 0											
Cloc	k cycle:6											
					ROB							
	BUS	 Y INSTR	UCTION	STATE	DEST	INATION	VAL	.UE				
ROB0	yes		1 1(R0)	MEM		R1	-					
ROB1	,		(/	EX		R2	-					
ROB2	•	LD R	3 3(R0)	EX		R3	-					
ROB3			-	-		-	-					
ROB4 ROB5			-	-		-	-					
ROB6			<u>-</u> _	_		_	_					
ROB7			_	-		_	_					
ROB8			-	_		_	_					
ROB9			-	-		-	-					
				INTEGE	R ADDE	R RS						
BUSY	OP	DEST	Vj VI	· Qj	Qk							
	- 	- 		- 	- 							
			IN		MULTIPL 	KS						

BUSY -			ST V	′j Vk 	-	Qk -				
				L	 OAD S	TORE QI	 JEUE 			
DEST ROB0 ROB1 ROB2	LD LD	-	-		- - -	1 2 3	4 8 -	- - -	-	
						NG TABL	 .E 			
0 L 4 L	D R1 1(F	₹0) ₹0)	1	2 4	3 5	4	MEM_F 7 - -	-	COMMIT_S - - -	COMMIT_F - - -
					INTE	GER AR	 F			
R0 - R3	31 -> 0									
					INTE	GER RA	т			
R1 -> R	OB0, R2	-> ROI	B1, R3 -	> ROB2						
Clock	cycle:	<u>:7</u>								
						ROB				
ROB0 ROB1 ROB2 ROB3	•	es es es	NSTRU LD R1 1 LD R2 2 LD R3 3	1(R0) 2(R0)	STATE MEM EX EX		INATION R1 R2 R3	VAL - - - -	UE	

R0 - R3	1 -> 0									
					INTI	EGER AR	F			
8 L	.D R3 3(F	₹0)	3	6	7	-		-		-
4 L	D R2 2(F	R0)	2	4	5	-		-	-	-
0 L	D R1 1(F	R0)	1	2	3	4	7	-	COMMIT_S	COMMIT_F
					TIMI	NG TABI	.E 			
ROB2	LD 	- 	- 	0 	- 	3	12 		 	
ROB1	LD	-	-	0	-	2	8			
DEST ROB0	TYPE LD	VSD -	QSD -	VAddr 0	QAd -	dr CON 1	ST ADE 4	νK V	AL FWD 	
						TORE Q				
-	- 	- 	- 	- 	- 	-				
BUSY	OP	DES	 T Vj	Vk	Qj	Qk				
				INT	EGER	MULTIPL	IER RS			
BUSY -	OP -	DES -	T Vj -	Vk -	Qj -	Qk -				
				 	NTEGE	R ADDE	R RS			
ROB9	no)	-		-		-	-		
ROB8	no		-	-			-	-		
ROB7	no		-	-			-	-		
ROB5 ROB6	no no		-		-		-	-		
ROB4	no		-		-		-	-		

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

Clock cycle:8

ROB9

no

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	-	-	-	-

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 0 4 8	INSTRUCTION LD R1 1(R0) LD R2 2(R0) LD R3 3(R0)	1 2 2 4	3	4	7	8	COMMIT_S	COMMIT_F - - -					
			INTE	GER ARF	 :								
R0 -	R31 -> 0												
			INTE	GER RAT	 「								
<u>Clo</u>	<u>ck cycle:9</u> 			ROB									
	BUSY	INSTRUCTION	STATE	DESTIN	 NATION	VAL	 UE						
ROB	o no	-	-		-	-							
ROB	•	LD R2 2(R0)			R2	-							
ROB	•	LD R3 3(R0)	EX		R3	-							
ROB	•	LD R4 4(R0)	ISSUE	. F	R4	-							
ROB		-	-	•	-	-							
ROB ROB		-	-	·	- _	_							
ROB		-	_		-	_							
ROB		_	_		_	_							
ROB		-	-		-	-							
			INTEGE	R ADDER	 R RS								
BUS	Y OP	 DEST Vj \	 /k Qj	 Qk									

-	-	-		 	-	-							
INTEGER MULTIPLIER RS													
BUS	Y OP	DES	ST V	 ′j Vk	Qj	Qk							
	- 	- 		 	- 	- 							
				L:	OAD S'	TORE QU	EUE 						
DES ⁻	T TYPE	VSD	QSD	VAddr	QAdd	dr CONS	T ADD	R VAI	_ FWD				
ROB	1 LD	-	-	0	-	2	8	-	-				
ROB		-	-	0	-	3	12	-	-				
ROB	3 LD	-	-	0	-	4	-	-	-				
	TIMING TABLE												
РС	INSTRUC	CTION	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB (COMMIT_S	COMMIT_F			
0	LD R1 1		1	2				8	9	9			
4	LD R2 2		2			8	11	-	-	-			
8	LD R3 3		3	6	7	-	-	-	-	-			
12	LD R4 4	(RU) 	9 	- 	- 	- 	-	- 	- 	-			
					INTE	GER ARI	=						
R1 ->	 > 5												
					INTE	GER RA	Γ						
R2 ->	 > ROB1, R	3 -> RO	B2, R4 -	> ROB3									
Clo	ck cycle	e:10											

ROB												
	BUS	Y II	NSTRUC	TION	STATE	DESTINA	TION	VALUE				
ROB0	no		-		-	-		-				
ROB1	yes		LD R2 2		MEM	R2		-				
ROB2	yes	S	LD R3 3	(R0)	EX	R3		-				
ROB3	yes	S	LD R4 4	(R0)	EX	R4		-				
ROB4	yes	s S	SUB R2 I	R1 R2	ISSUE	R2		-				
ROB5	no		-		-	-		-				
ROB6	no		-		-	-		-				
ROB7	no		-		-	-		-				
ROB8	no		-		-	-		-				
ROB9	no		-		-	-		-				
INTEGER ADDER RS												
BUSY	OP	DES	•		Qj	Qk						
yes 	SUB	ROE	34 5 	- 	- 	ROB1						
INTEGER MULTIPLIER RS												
BUSY -	OP -	DES -	T Vj -	Vk -	Qj -	Qk -						
				L(OAD ST	ORE QUE	 JE 					
DEST	TYPE	VSD	OSD	VAddr	OAddr	CONST	ADDR	\/ AI	FWD			
ROB1	LD	-	-	0		2	8	-	-			
	LD	_	_	0	_	3	12	_	_			
	LD	-	-	0	-	4	-	-	-			
	TIMING TABLE											
0 I	ISTRUCTI LD R1 1(R LD R2 2(R	.0)	SSUE 1 2	EX_S 2 4	EX_F N 3 5	ИЕМ_S М 4 8	7	VB CC 8 -	DMMIT_S 9 -	COMMIT_F 9 -		

8 12 16	LD R3 3(R0) LD R4 4(R0) SUB R2 R1 R	Ū	6 10 -	7 11 -	 	- - -	- - -	- - -							
	INTEGER ARF														
R1 ->	R1 -> 5														
				INTEG	ER RAT										
R2 ->	R2 -> ROB4, R3 -> ROB2, R4 -> ROB3														
Cloc	Clock cycle:11														
	 ROB														
	BUSY	INSTRU	CTION	STATE	DESTINATIO	N VALUE									
ROB(-	o (Do)	-	-	-									
ROB2	,	LD R2		MEM	R2	-									
ROB3	•	LD R3 LD R4		EX	EX R3 - EX R4 -										
ROB4	•	SUB R2		ISSUE	R2	_									
ROB	•	BEQ R2		ISSUE	36	_									
ROB6	•	_		-	-	_									
ROB7	7 no	-		-	-	-									
ROB8	3 no	-		-	-	-									
ROBS	9 no	-		-	-	-									
				INTEGER	ADDER RS										
BUSY yes yes	SUB R		5 -	Qj - ROB4	Qk ROB1 ROB3										
			 L	 -OAD ST(ORE QUEUE										

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

	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	_	_	_	_

ROB	39 no										
	INTEGER ADDER RS										
BUS' no yes	SUB	DEST ROB4 ROB5	5	1.0	Vk Qj Qk 1.0 - ROB4 ROB3						
				L(OAD ST	ORE QU	 EUE 				
DEST ROBS			QSD - -	VAddr 0 0	-	r CONS 3 4	12	-	AL FWD - -		
	TIMING TABLE										
PC 0 4 8 12 16 20	INSTRUCT LD R1 1(F LD R2 2(F LD R3 3(F LD R4 4(F SUB R2 F BEQ R2 F	R0) R0) R0) R0) R1 R2	SSUE 1 2 3 9 10	2 4 6	3	4 8	7	8	COMMIT_S 9 - - - -	COMMIT_F 9 - - - -	
					INTE	GER ARF	:				
R1 ->	> 5 										
	INTEGER RAT										
R2 ->	R2 -> ROB4, R3 -> ROB2, R4 -> ROB3										
Clo	ck cycle:	<u>13</u> 									

	BUS	SY I	NSTRU	CTION	STATE	E DEST	INATION	VAL	_UE			
ROB	0 no)	-		-		-	-				
ROB	1 no)	-		-		-	-				
ROB:	,			3(R0)			R3	-				
ROB	,			ł(R0)			R4	-				
ROB	•			R1 R2			R2	-				
ROB	,		BEQ R2	R4 3	ISSUE	Ξ	36	-				
ROB			-		-		-	-				
ROB			-		-		-	-				
ROB			-		-		-	-				
ROB	9 no)	-		-		-	-				
INTEGER ADDER RS												
BLIC	 Y OP	DES	 ST V	 ′j Vk	Qi	Qk						
		ROB		-	•	- -						
yes		ROB				34 ROB	3					
LOAD STORE QUEUE												
DES ⁻	T TYPE	VSD	QSD	\/∆ddr	. 044	dr CON	ST ADD	D V	AL FWD			
ROB:		-	- -	0	<u>ي</u> -	3	12					
ROB		-	_	0	-	4	16					
					TIMI	NG TABL	.E 					
5.0	NIOTD:::=								001			
PC									COMMIT_S			
0	LD R1 1(F	=							9	9		
4 8	LD R2 2(F LD R3 3(F						11 15		13 -	13		
o 12	LD R3 3(F LD R4 4(F						15 -			-		
16	SUB R2 F			13			-			- -		
	BEQ R2 F						_	-	-	-		
	INTEGER ARF											

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 TARLE										

TIMING TABLE

РС	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

	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	-	-	-	-	

ROE ROE		no no		-		-		-	-			
INTEGER ADDER RS												
BUS yes		OP BEQ	DE: ROI	ST V 35 4	/j Vk .0 -	Qj						
LOAD STORE QUEUE												
	DEST TYPE VSD QSD VAddr QAddr CONST ADDR VAL FWD ROB3 LD 0 - 4 16											
	TIMING TABLE											
PC 0 4 8 12 16 20	L L L	STRUCT .D R1 1(F .D R2 2(F .D R3 3(F .D R4 4(F SUB R2 F SEQ R2 F	R0) R0) R0) R0) R1 R2	ISSUE 1 2 3 9 10 11	2 4 6 10	3	4 8	7	8	COMMIT_S 9 13 - - -	COMMIT_F 9 13 - - -	
						INT	EGER AR	 F				
R1 -	> 5,	, R2 -> 1										
						INT	EGER RA	Т				
R2 -	R2 -> ROB4, R3 -> ROB2, R4 -> ROB3											
<u>Clo</u>	Clock cycle:17											
							ROB					

	BUSY	INSTRUC	TION	STATE	DEST	INATION	VAL	UE	
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	Ē	36	-		
ROB6	no	-		-		-	-		
ROB7	no	-		-		-	-		
ROB8	no	-		-		-	-		
ROB9	no	-		-		-	-		
			I	NTEGE	R ADDE	R RS			
BUSV	OP DE	 -ST Vi	\ \/k	 : Qi	Qk				
	BEQ RO	-		-	ROB3				
			<i>-</i>						
			L	OAD S	TORE QI	JEUE			
DEST	TYPE VSI	D QSD	VAddr	QAd	dr CON	ST ADD	R V	AL FWD	
ROB3	LD -	-	0	-	4	16	-	-	
				TIMI	 NG TABI	 .E			
PC IN	ISTRUCTION	ISSUE	EX S	EX F	MEM S	MEM_F	WB	COMMIT S	COMMIT_F
	LD R1 1(R0)	1	2	3	4	7	8	9	9
	LD R2 2(R0)		4			11	12	13	13
	LD R3 3(R0)	3		7		15			17
	LD R4 4(R0)	9	10			19	-	-	_
	SUB R2 R1 R2					-		_	_
	BEQ R2 R4 3								-
					GER AR				
R1 -> 5	, R2 -> 1								

			114120				
R2 -> R	ROB4, R4 -> F	 ROB3					
Clock	cycle:18						
			F	ROB			
	BUSY	INSTRUCTION	STATE	DESTINATIO	N 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 yes		EST Vj V OB5 4.0 -	•	Qk ROB3			
			LOAD ST	ORE QUEUE			
	TYPE VS LD -	SD QSD VAdd - 0	r QAddr -	CONST A	DDR VAL 16 -	FWD -	
			TIMIN	G TABLE			
0 l 4 l	ISTRUCTION LD R1 1(R0) LD R2 2(R0) LD R3 3(R0)	1 2	3	MEM_S MEM_ 4 7 8 11 12 15	_F WB C0 8 12 16	OMMIT_S 9 13 17	COMMIT_F 9 13 17

12 16	LD R4 4(R0) SUB R2 R1 R	9	10 13	11 13	16 -	19 -	- 14	-	- -
20	BEQ R2 R4 3		-	-	-	-	-	-	-
				INTE	GER AR	 F			
R1 ->	5, R2 -> 1, R3	->1							
				INTE	GER RA	.T			
R2 ->	ROB4, R4 -> F	ROB3							
Cloc	<u>:k cycle:19</u> 								
					ROB				
	BUSY	INSTF	RUCTION	I STATE	DEST	INATION	VALUE		
ROB0) no		-	-		-	-		
ROB1			-	-		-	-		
ROB2			-	-		-	-		
ROB3	•		4 4(R0)	MEM		R4	-		
ROB4			R2 R1 R2			R2	4.0		
ROB5	•	BEQ	R2 R4 3	ISSUE		36	-		
ROB6			-	-		-	-		
ROB7			-	-		-	-		
ROB8			-	-		-	-		
ROB9) no		-	-		-	-		
				INTEGE	R ADDE	R RS			
BUSY	 ′ OP [DEST	Vj '	√k Qj	Qk				
yes	BEQ R	OB5	4.0		ROB3				
				LOAD S	ΓORE QI	JEUE			

DEST TYPE VSD QSD VAddr QAddr CONST ADDR VAL FWD

ROB	3 L	D -	-	0	-	4	16	-	-	
					TIMI	NG TABL	.E			
PC	INICTE	RUCTION	ISSUE	EV 8		MEM S	NAENA E	\A/D	COMMIT S	COMMIT E
0		11 1(R0)	1550E	2	EX_F 3	WEW_5 4	MEM_F 7	WB 8	9	COMMIT_F 9
4		2 2(R0)	2			8	, 11	40	13	13
8		3 3(R0)	3	6		12	15		17	17
12		84 4(R0)	9		11	4.0	19	-	-	-
16		R2 R1 R2					-		-	_
20		R2 R4 3		-	-	-	-	-	-	-
					INTE	GER AR	 F			
 R1 ->	> 5, R2	-> 1, R3->	 1							
					INTE	GER RA	 Т			
		 1, R4 -> R0 	DB3							
<u>Clo</u>	<u>ck cy</u> 	cle:20								
						ROB				
		BUSY	INSTRUC		STATE		NATION	VAL	 UE	
ROB	0	no	-		-		-	-		
ROB		no	-		-		-	-		
ROB		no	-		-		-	-		
ROB		no	LD R4 4		WB		R4	0.		
ROB		no	SUB R2		WB		R2	4.	0	
ROB		yes	BEQ R2	R4 3	ISSUE	=	36	-		
ROB		no	-		-		-	-		
ROB		no	-		-		-	-		
ROB		no	-		-		-	-		
ROB	9	no	-		-		-	-		

INI	TEGEI		NED	PC
117	IEGEI	T AL	リノヒス	c

	DEST ROB5	•	•	

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

	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 ROB9	no no	-		-		-	-		
			l	INTEGE	R ADDE	R RS			
BUSY no	OP BEQ F		Vj Vk 4.0 0.0	-	Qk -				
				TIMII	NG TABL	 .E 			
0 4 8 12 16	NSTRUCTIO LD R1 1(R0) LD R2 2(R0) LD R3 3(R0) LD R4 4(R0) SUB R2 R1 I BEQ R2 R4	1 2 3 9 R2 10	2	3 5	4	MEM_F 7 11 15 19 -	WB 8 12 16 20 14	COMMIT_S 9 13 17 21 -	COMMIT_F 9 13 17 21 -
				INTE	GER AR	F 			
R1 -> 5	5, R2 -> 1, R3	3->1 							
				INTE	GER RA	Т			
R2 -> F	ROB4								
<u>Clock</u>	k cycle:22	<u>)</u>							
					ROB				
ROB0 ROB1 ROB2 ROB3	BUSY no no no no	INSTRU - - - -	JCTION	STATE - - - - -	DESTI	 NATION - - -	VAL - - - -	.UE	

ROE ROE ROE ROE ROE	35 36 37 38	no yes yes no no no	- BEQ R2 MULT R' - - -		- EX - - - -		- 36 - - - -	- - - - -		
				 	NTEGE	R ADDEI	R RS			
					TIMI	 NG TABL	 .E 			
PC 0 4 8 12 16 20 24	LD R1 LD R2 LD R3 LD R4 SUB R	2(R0) 3(R0) 4(R0) 22 R1 R2 22 R4 3	1 2 3 9	EX_S 2 4 6 10 13 21	EX_F 3 5 7 11 13 21	MEM_S 4 8 12 16 - -	MEM_F 7 11 15 19 - -	WB 8 12 16 20 14	COMMIT_S 9 13 17 21 22 -	COMMIT_F 9 13 17 21 22 -
					INTE	EGER AR	F 			
R1 -	-> 5, R2 ->	> 4, R3-> 	·1,R4->0 							
					INTE	EGER RA	т 			
R1 -	-> ROB6									
Clo	ock cyc	le:23				ROB				
	 !	 BUSY	INSTRU	CTION	STATE	E DESTI	NATION	 VAI	 -UE	

ROB0	no	-		-		-	-		
ROB1	no	-		-		-	-		
ROB2	no	-		-		-	-		
ROB3	no	-		-		-	-		
ROB4	no	-		-		-	-		
ROB5	no	-		-		-	-		
ROB6	•	MULT R		EX		R1	-		
ROB7	yes	SUB R2	R2 R3	ISSUE		R2	-		
ROB8	no	-		-		-	-		
ROB9	no	-		-		-	-		
			 I	NTEGE	R ADDE	R RS			
BUSY no			/j Vk 4 1	•	Qk -				
				TIMI	NG TABI	 -E			
PC INS	STRUCTION	ISSUE	FY S	FY F	MEM S	MEM_F	WR	COMMIT_S	
	D R1 1(R0)	1	2	3	4	7	8	9	9
	D R2 2(R0)	2	4	5	8	11	12	13	13
	D R3 3(R0)	3	6	7	12	15	16	17	17
	D R4 4(R0)	9	10	11	16	19	20	21	21
	UB R2 R1 R2		13	13	_	-	14	22	22
	EQ R2 R4 3	11	21	21	_	-	22	23	23
	_T R1 R1 R2	22	23	23	-	_	-	-	-
28 SUE	3 R2 R2 R3	23	-	-	-	-	-	-	-
				INTE	GER AR	 RF			
R1 -> 5,	R2 -> 4, R3->	1, R4->0							
				INTE	 EGER RA	 \T			
R1 -> R0	 DB6, R2 -> R0)B7							

Clock cycle:24

					ROB				
	BUSY	INSTRU	CTION	STATE	DEST	NATION	VAL	 UE	
ROB0	no	-		-		-	-		
ROB1	no	-		-		-	-		
ROB2	no	-		-		-	-		
ROB3			-		-	-			
ROB4 no -		-		-	-				
ROB5 no -		-		-	-				
ROB6 yes MULT R1 R1 R2		1 R1 R2	EX		R1	20)		
ROB7	yes	SUB R2				R2	-		
ROB8	yes	BNE R2	R3 -1	ISSUE		24	-		
ROB9	no	-		-		-	-		
				NTEGE	R ADDE	R RS			
BUSY no yes	SUB	EST ROB7 ROB8	Vj V 4 1 - 1	-	-				
				TIMIN	NG TABL	.E			
	OTDUOTION	100115			N45N4 O				
	STRUCTION D R1 1(R0)	1550E	ΕΧ_S 2	ΕΧ_F 3		7	8 8	9	
	` ,	2	4	5 5	4 8	, 11	o 12	9 13	9 13
	D R2 2(R0) D R3 3(R0)	3	6	5 7	12	15	16	13 17	13 17
	D R4 4(R0)	9	10	, 11	16	19	20	21	21
			13	13	-	-	14	22	22
12 L	IIR R2 R1 R2	, 111		117		_	17	~~	
12 L 16 S	SUB R2 R1 R2 SEO R2 R4 3				_	_	22		
12 L 16 S 20 B	EQ R2 R4 3	11	21	21	-	-	22 24		23
12 L 16 S 20 B 24 M		11 R2 22			- - -	- - -	22 24 -		

			_	_	_		_	
ı	N	ГΕ	G	E	к	Α	к	H

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

INTEGER RAT

R1 -> ROB6, R2 -> ROB7

Clock cycle:25

RO	В
----	---

	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 20 24 28 32	SUB R2 R1 F BEQ R2 R4 3 MULT R1 R1 SUB R2 R2 R BNE R2 R3 -3	3 11 R2 22 3 23	13 21 23 24 -	13 21 23 24 -	- - - -	- - - -	14 22 24 25 -	22 23 - -	22 23 - - -					
	INTEGER ARF													
R1 -> 20, R2 -> 4, R3->1, R4->0														
				INTEC	GER RAT									
R2 -:	> ROB7													
Clo	ck cycle:26 			 F	ROB									
	BUSY	INSTR	 LICTION	STATE	DESTINA	 TION	 \/ΔΙΙΙΕ							
ROB		1110111	-	-	-	11011	-							
ROB			_	_	_		_							
ROB			_	_	_		_							
ROB			_	_	_		_							
ROB			_	_	_		-							
ROB			_	_	_		-							
ROB			-	_	_		-							
ROB			-	-	-		-							
ROB		BNE R	2 R3 -1	EX	24		-							
ROB9 no														
	INTEGER ADDER RS													
BU no		DEST ROB8	Vj V 3 1		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	2 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

no

no

no

yes

INTEGER RAT

R0 - R31 -> R0 - R31

Clock cycle:27

ROB6

ROB7

ROB8

ROB9

ROB											
	BUSY	INSTRUCTION	STATE	DESTINATION	VALUE						
ROB0	no	-	-	-	-						
ROB1	no	-	-	-	-						
ROB2	no	-	-	-	-						
ROB3	no	-	-	-	-						
ROB4	no	-	-	-	-						
ROB5	no	-	-	-	-						

MULT R1 R1 R2 ISSUE

INTEGER ADDER RS

Вι	JSY	OP	DE	ST \	/j Vk	. Qj	Qk				
					INT	EGER	MULTIPL	IER RS			
BUS N		OP MULT	DES ROE	-	Vk 0 3	 Qj -	Qk -				
						TIMI	NG TABL	 .E			
24	LC LC LC SU BE MI SU BN	TRUCTION R1 1(R) R2 2(R) R3 3(R) R4 4(R) JB R2 R ULT R1 JB R2 R NE R2 R ULT R1 ULT R1	0) 0) 0) 0) 1 R2 4 3 R1 R2 2 R3 3 -3 R1 R2	1 2 3 9 10 11 22 23 24 27	2 4 6 10 13 21 23 24 26	3 5 7 11 13 21 23 24 26	MEM_S 4 8 12 16 EGER AR	7 11 15 19 - - - - -	WB 8 12 16 20 14 22 24 25 27	COMMIT_S 9 13 17 21 22 23 25 26 27	COMMIT_F 9 13 17 21 22 23 25 26 27 -
						INTI	EGER RA	 Т			
 R1 -	> RC)B9									
<u>Clo</u>	ck (cycle:2	<u>28</u>								
							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 LD R1 1(R0) LD R2 2(R0) LD R3 3(R0) LD R4 4(R0) **SUB R2 R1 R2 BEQ R2 R4 3** MULT R1 R1 R2 22 **SUB R2 R2 R3** BNE R2 R3 -3 MULT R1 R1 R2 27 **SUB R2 R2 R3**

					INTEG	ER ARF				
R1 -> 20,	R2 -> 3,	R3->1, R	4->0							
					INTEG	ER RAT				
R1 -> RO	B9, R2 ->	ROB0								
Clock	cycle:29	<u>9</u>								
					R	ЮВ				
ROB0 ROB1 ROB2 ROB3 ROB4 ROB5 ROB6 ROB7 ROB8	BUSY yes yes no no no no no no	SUB	FRUCT RUCT RE R2 R	2 R3	STATE EX ISSUE - - - - - -	DESTINATIC R2 24 - - - - - -				
ROB9	no	MULT F	R1 R1	R2	WB	R1	60			
				I	NTEGER	ADDER RS				
BUSY no yes		DEST ROB0 ROB1	3	1	Qj - 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	2 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

	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 ROB9	no no		-		-		-	-						
	INTEGER ADDER RS													
BUSY OP DEST Vj Vk Qj Qk no BNE ROB1 2 1														
	INTEGER MULTIPLIER RS													
BUSY	BUSY OP DEST Vj Vk Qj Qk													
	TIMING TABLE													
0 4 8 12 16 20 24 28 32 24 28	NSTRUCTION LD R1 1(RO LD R2 2(RO LD R3 3(RO LD R4 4(RO SUB R2 R1 BEQ R2 R4 MULT R1 F SUB R2 R2 BNE R2 R3 MULT R1 F SUB R2 R3	0) 0) 0) 0) R2 4 3 R1 R2 2 R3 2 R3 2 R3	SUE 1 2 3 9 10 11 22 23 24 27 28	EX_S 2 4 6 10 13 21 23 24 26 28 29	EX_F 3 5 7 11 13 21 23 24 26 28 29	MEM_S 4 8 12 16	MEM_F 7 11 15 19	WB 8 12 16 20 14 22 24 25 27 29 30	COMMIT_S 9 13 17 21 22 23 25 26 27 30 31	COMMIT_F 9 13 17 21 22 23 25 26 27 30 31				
					INTE	EGER AR								
R1 -> 6	R1 -> 60, R2 -> 3, R3->1, R4->0													
INTEGER RAT														
R1 -> F	ROB9, R2 ->	> ROB0				_								

Clock cycle:31

						ROB					
	BUSY	INS	TRUC	TION	STATE	DEST	NATION	VAL	_UE		
ROB0	no		-		-		-	-			
ROB1	yes	BN	ER2	₹3 -3	ISSUE	Ξ	24	-			
ROB2	no		-		-		-	-			
ROB3	no		-		-		-	-			
ROB4	no		-		-		-	-			
ROB5	no		-		-		-	-			
ROB6	no		-		-		-	-			
ROB7	no		-		-		-	-			
ROB8	no		-		-		-	-			
ROB9	no		-		-		-	-			
INTEGER ADDER RS											
BUSY no	OP BNE	DEST ROB1	Vj 2	Vk 1	Qj -	Qk -					
				INT	EGER	MULTIPL	IER RS				
BUSY	OP I	DEST	Vj	Vk	Qj	Qk					
					TIMI	NG TABL	 .E 				
0 LE 4 LE 8 LE 12 LE 16 SU 20 BE 24 MU	TRUCTIO) R1 1(R0)) R2 2(R0)) R3 3(R0)) R4 4(R0) JB R2 R1 EQ R2 R4 JLT R1 R2) 1) 2) 3) 9 R2 1 3 1	1 2 3 9 0 1	EX_S 2 4 6 10 13 21 23 24	EX_F 3 5 7 11 13 21 23 24	MEM_S 4 8 12 16	MEM_F 7 11 15 19 - - -	WB 8 12 16 20 14 22 24 25	COMMIT_S 9 13 17 21 22 23 25 26	COMMIT_F 9 13 17 21 22 23 25 26	

32 24 28 32	BNE R2 R3 MULT R1 R SUB R2 R2 BNE R2 R3	1 R2 27 R3 28	26 28 29 31	26 28 29 31	 		27 30 31 -	27 30 31 -						
	INTEGER ARF													
	60, R2 -> 2, I													
				INTEC	SER RAT									
R1 - F	R1 - R31 -> R1 - R31													
Cloc	Clock cycle:32													
	ROB													
	BUSY	INSTRI	JCTION	STATE	DESTINATIO	N VALUE								
ROBO		-	<u>-</u>	-	-	-								
ROB1		-	-	-	-	-								
ROB2	•	MULT F	R1 R1 R2	ISSUE	R1									
ROB3		-	į	-	-	-								
ROB4		-	•	-	-	-								
ROB5		-		-	-	-								
ROB6		-	-	-	-	-								
ROB7	_	-	•	-	-	-								
ROB8		-	•	-	-	-								
ROB9) no			-	-	-								
	INTEGER ADDER RS													
BUS	BUSY OP DEST Vj Vk Qj Qk													
	INTEGER MULTIPLIER RS													

BUS no		DEST ROB2	Vj Vk 60 2	Qj -	Qk -								
	TIMING TABLE												
PC 0 4 8 12 16	INSTRUCT LD R1 1(R LD R2 2(R LD R3 3(R LD R4 4(R SUB R2 R	20) 20) 20)	SUE EX_S 1 2 2 4 3 6 9 10 10 13	3 5	MEM_S 4 8 12 16	MEM_F 7 11 15 19	8 12 16 20	COMMIT_S 9 13 17 21 22	COMMIT_F 9 13 17 21 22				
20 24 28 32 24 28 32	BEQ R2 R MULT R1 SUB R2 R BNE R2 R MULT R1 SUB R2 R BNE R2 R	A 3 R1 R2 2 R3 3 -3 2 R1 R2 2 R3	11 21 22 23 23 24 24 26 27 28 28 29 29 31	21 23 24 26 28 29 31	- - - - -	-	22 24 25 27 29 30 32	23 25 26 27 30 31 32	23 25 26 27 30 31 32				
 R1 -	 > 60, R2 -> 2	 2, R3->1,	 R4->0	INTE	GER AR	F 							
				INTE	GER RA	 Т							
R1 -:	 > ROB2												
Clo	ck cycle::	<u>33</u>											
					ROB								
ROB ROB ROB ROB	0 no 1 no 2 ye 3 ye	s MU s SU	STRUCTION LT R1 R1 R2 JB R2 R2 R3 -	-	I	NATION R1 R2 -	VAL - - - -	UE					

ROB5 ROB6 ROB7 ROB8 ROB9	no no no		- - - -	- - - -		- - - -	- - - -					
BUSY	 OP	DEST			R ADDE	R RS 						
no 	SUB 	ROB3	2 1		-							
	INTEGER MULTIPLIER RS											
BUSY no	OP MULT	DEST ROB2	Vj VI 60 2	-	Qk -							
	TIMING TABLE											
0	INSTRUCTIC LD R1 1(R0) 1	2	EX_F	MEM_S	MEM_F	8	COMMIT_S 9	9			
4 8	LD R2 2(R0) LD R3 3(R0)		4 6	5 7	8 12	11 15	12 16	13 17	13 17			
12	LD R4 4(R0)		10	11	16	19	20	21	21			
16 20	SUB R2 R1 BEQ R2 R4		13 21	13 21	-	-	14 22	22 23	22 23			
24	MULT R1 R		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 R		28	28	-	-	29	30	30			
28	SUB R2 R2		29	29	-	-	30	31	31			
32	BNE R2 R3		31	31	-	-	32	32	32			
24 28	MULT R1 R SUB R2 R2		33	33 -	-	-	-	-	-			
				INTI	EGER AR	 F 						

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 R	OB4	1 1	-	-								
	TIMING TABLE												
PC	INSTRUCTION	ISSUE	EX_S	EX F	MEM_S	MEM_F	WB	COMMIT_S					
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 R		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 R	3 23	24	24	-	-	25	26	26				
32	BNE R2 R3 -3	3 24	26	26	-	-	27	27	27				
24	MULT R1 R1	R2 27	28	28	-	-	29	30	30				
28	SUB R2 R2 R	3 28	29	29	-	-	30	31	31				
32	BNE R2 R3 -3	3 29	31	31	-	-	32	32	32				
24	MULT R1 R1	R2 32	33	33	-	-	34	35	35				
28	SUB R2 R2 R	3 33	34	34	-	-	35	36	36				
32	BNE R2 R3 -	3 34	-	-	-	-	-	-	-				
				INTE	EGER AR	 F							
 R1 -	> > 120, R2 -> 2, F	 R3->1, R4-	>0										
				INTF	 EGER RA	 T							
R2 -	> ROB3												
Clo	ck cycle:36												
					ROB								
	BUSY INSTRUCTION STATE DESTINATION VALUE												
ROB	30 no	-		-		-	-						
ROB	31 no	-		-		-	-						

ROB2		-		-		-	-		
ROB3	3 no	-		-		-	-		
ROB4	4 yes	BNE R	2 R3 -3	EX		24	-		
ROB	5 no	-		-		-	-		
ROB6		-		-		-	-		
ROB7		-		-		-	-		
ROB	8 no	-		-		-	-		
ROB9	9 no	-		-		-	-		
				INTEGE	R ADDE	 R RS			
BUSY	 Y OP DE	ST '	 Vj VI	 k Qi	 Qk				
no	BNE RC	B4	1 1	-	-				
				TIMI	NG TABL	 .E			
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
	MULTIDADADA	000	23	23		_	24	25	25
24	MULT R1 R1 R	2 22	23	23	-	-	27	20	25
24 28	SUB R2 R2 R3	2 22 23	24	24	-	-	25	26	26
		23 24			- -				

INTEGER ARF

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

SUB R2 R2 R3

BNE R2 R3 -3

SUB R2 R2 R3

BNE R2 R3 -3

MULT R1 R1 R2 32

Clock cycle:37

ROB											
	BUSY	INSTRU	CTION	STATE	DESTI	NATION	VAL	JE			
ROBO) no	-		-		-	-				
ROB1	1 no	-		-		-	-				
ROB2	2 no	-		-		-	-				
ROB3	3 no	-		-		-	-				
ROB4	1 no	-		-		-	-				
ROB5	5 yes	SD R1	5(R0)	ISSUE	Ē	-	-				
ROB6	6 no	-		-		-	-				
ROB7		-		-		-	-				
ROB8	3 no	-		-		-	-				
ROBS) no	-		-		-	-				
			L	OAD S	TORE QU	JEUE					
DEST ROB5			VAdd 0	r QAd -	dr CON	ST ADD	R VA -	L FWD -			
				TIMI	NG TABL	.E					
	INSTRUCTION							COMMIT_S			
0	LD R1 1(R0)	1	2			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		13	13	-	-	14	22	22		
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23		
24	MULT R1 R1 R		23	23	-	-	24	25	25		
28	SUB R2 R2 R3		24	24	-	-	25	26	26		
32	BNE R2 R3 -3	24	26	26	-	-	27	27	27		
24	MULT R1 R1 R		28	28	-	-	29	30	30		
28	SUB R2 R2 R3		29	29	-	-	30	31	31		
32	BNE R2 R3 -3	29	31	31	-	-	32	32	32		
24	MULT R1 R1 R		33	33	-	-	34	35	35		
28	SUB R2 R2 R3	33	34	34	-	-	35	36	36		

32 36	BNE R2 R3 -3 SD R1 5(R0)	34 37	36 -	36 -	- -	- -	37 -	37 -	37 -				
	INTEGER ARF												
R1 ->	R1 -> 120, R2 -> 1, R3->1, R4->0												
Clo	Clock cycle:38												
				i	ROB								
	BUSY	INSTRU	CTION	STATE	DESTI	NATION	VAL	JE					
ROB		-		-		-	-						
ROB ²		-		-		-	-						
ROB2		-		-		-	-						
ROB3		-		-		-	-						
ROB!		SD R1	5(P0)	ĒΧ	2	- 20	_						
ROB	,	- JD IXI .	J(110)	-	2	-	_						
ROB		_		_		_	_						
ROB		_		-		_	_						
ROB	9 no	-		-		-	-						
			L	 OAD ST	ORE QU	 IEUE							
DEST	TYPE VSI	D QSD	VAddr	QAddı	r CONS	ST ADD	R VA	L FWD					
ROB!	5 SD 120) -	0	-	5	-	-	-					
				TIMIN	G TABL	E							
РС	INSTRUCTION	ISSUE	EX_S	FX 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		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	2 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 M	MEM_F \	NB CO	MMIT_S C	OMMIT_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 ROBO 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

				L	OAD S	TORE QU	JEUE			
DEST ROB5		VSD 120	QSD -	VAddr 0	· QAdo	dr CONS	ST ADD 20		AL FWD	
					 11MIT	NG TABL	E			
PC	INSTRUCTI	ON	ISSUE	EX_S	EX_F	MEM_S	MEM_F	WB	COMMIT_S	COMMIT_F
0	LD R1 1(R	0)	1	2	3	4	7	8	9	9
4	LD R2 2(R	•	2	4	5	8	11	12	13	13
8	LD R3 3(R	•	3	6	7	12	15	16	17	17
12	LD R4 4(R	•	9	10	11	16	19	20	21	21
16	SUB R2 R		10	13	13	-	-	14	22	22
20	BEQ R2 R		11	21	21	-	-	22	23	23
24	MULT R1			23	23	-	-	24	25	25
28	SUB R2 R		23	24	24	-	-	25	26	26
32	BNE R2 R		24	26	26	-	-	27	27	27
24	MULT R1			28	28	-	-	29	30	30
28	SUB R2 R		28	29	29	-	-	30	31	31
32	BNE R2 R		29	31	31	-	-	32	32	32
24	MULT R1			33	33	-	-	34	35	35
28	SUB R2 R		33	34	34	-	-	35	36	36
32	BNE R2 R		34	36	36	-	-	37	37	37
36	SD R1 5(F	R0)	37	38	39	-	-	40	-	-
					INTE	GER AR	F			
 R1 ->	120, R2 ->	1, R3	 ->1, R4->	 >0						
Cloc	k cycle:4	<u> 11</u>								
						ROB				
	BUS	Υ	INSTRU	 CTION	STATE	DESTI	 NATION	VAL	 .UE	
ROBO) no		-		-		-	-		
	l no									

ROB	2 no)	_		_		-	-					
ROB	3 no)	-		-		-	-					
ROB-	4 nc)	-		-		-	-					
ROB	5 no)	-		-		-	-					
ROB	6 no)	-		-		-	-					
ROB	ROB7 no		-		-		-	-					
ROB	ROB8 no				-		-	-					
ROB	9 no)	-		-		-	-					
	LOAD STORE QUEUE												
DEST TYPE VSD QSD VAddr QAddr CONST ADDR VAL FWD													
ROB	5 SD	120	-	0	-	5	20	-					
	TIMING TABLE												
PC	INSTRUCT	ION	ISSUE	EX S	EX_F	MEM S	MEM_F	WB	COMMIT S	COMMIT_F			
0	LD R1 1(F		1	2	3	4	7	8	9	9			
4	LD R2 2(F	•	2		5	8	11	12	13	13			
8	LD R3 3(F	•	3	6	7	12	15	16	17	17			
12	LD R4 4(F	RO)	9	10	11	16	19	20	21	21			
16	SUB R2 R	R1 R2	10	13	13	-	-	14	22	22			
20	BEQ R2 F	R4 3	11	21	21	-	-	22	23	23			
24	MULT R1	R1 R2	2 22	23	23	-	-	24	25	25			
28	SUB R2 R	R2 R3	23	24	24	-	-	25	26	26			
32	BNE R2 R	23 -3	24	26	26	-	-	27	27	27			
24	MULT R1	R1 R2	2 27	28	28	-	-	29	30	30			
28	SUB R2 R	R2 R3	28	29	29	-	-	30	31	31			
32	BNE R2 R	23 -3	29	31	31	-	-	32	32	32			
24	MULT R1	R1 R2	2 32	33	33	-	-	34	35	35			
28	SUB R2 R	R2 R3	33	34	34	-	-	35	36	36			
32	BNE R2 F	₹3 -3	34	36	36	-	-	37	37	37			
36	SD R1 5(R0)	37	38	39	-	-	40	41	44			
					INT	EGER AR	 F						
 R1 ->	 > 120, R2 ->	1, R3-	 ->1, R4->	 >0									

Clock cycle:42

ROB											
	BUSY	INSTRU	CTION	STATE	DESTI	NATION	VAL	 UE			
ROB0	no	-		-		-	-				
ROB1	no	-		-		-	-				
ROB2		-		-		-	-				
ROB3		-		-		-	-				
ROB4		-		-		-	-				
ROB5		-		-		-	-				
ROB6		-		-		-	-				
ROB7		-		-		-	-				
ROB8		-		-		-	-				
ROB9	no	-		-		-	-				
			L	OAD S	TORE QU	JEUE					
DEST ROB5			VAddı 0	r QAd -	dr CON	ST ADD 20		 AL FWD -			
				ТІМІ	NG TABL	.E					
PC I	NSTRUCTION	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		13	13	-	-	14	22	22		
20	BEQ R2 R4 3	11	21	21	-	-	22	23	23		
24	MULT R1 R1 F		23	23	-	-	24	25	25		
28	SUB R2 R2 R3		24	24	-	-	25	26	26		
32	BNE R2 R3 -3		26	26	-	-	27	27	27		
24	MULT R1 R1 R		28	28	-	-	29	30	30		
28	SUB R2 R2 R3		29	29	-	-	30	31	31		
32	BNE R2 R3 -3		31	31	-	-	32	32	32		
24	MULT R1 R1 F		33	33	-	-	34 25	35	35		
28	SUB R2 R2 R3	3 33	34	34	-	-	35	36	36		

32 36	BNE R2 R3 -3 SD R1 5(R0)	34 37	36 38	36 39	-	- -	37 40	37 41	37 44				
	INTEGER ARF												
R1 ->	R1 -> 120, R2 -> 1, R3->1, R4->0												
	Clock evale 42												
<u>Clo</u>	Clock cycle:43												
					ROB								
	BUSY	INSTRU	CTION	STATE	DESTI	NATION	VAL	 UE					
ROB		-		-		-	-						
ROB		-		-		-	-						
ROB		-		-		-	-						
ROB ROB		-		-		-	-						
ROB		_		_		_	_						
ROB		_		_		_	_						
ROB		_		_		_	_						
ROB		-		-		_	_						
ROB	9 no	-		-		-	-						
			 I	 OAD ST	 ORE QU	 IFUF							
DES	T TYPE VSI	D QSD	VAddr	QAdd	r CONS	ST ADD	R V	AL FWD					
ROB	5 SD 120	0 -	0	-	5	20	-	-					
				TIMIN	IG TABL	E							
РС	INSTRUCTION	ISSUE	EX_S	FX 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		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	2 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.