Minor Exam Test Cases

COL216 : Minor Exam
II Semester 2020-2021

Dept. of Computer Science and Engineering
IIT Delhi

Release Time: March 20, 2021(Saturday), 2:00 PM
Deadline: March 21, 2021(Sunday), 11:59PM

Part 1: DRAM implementation

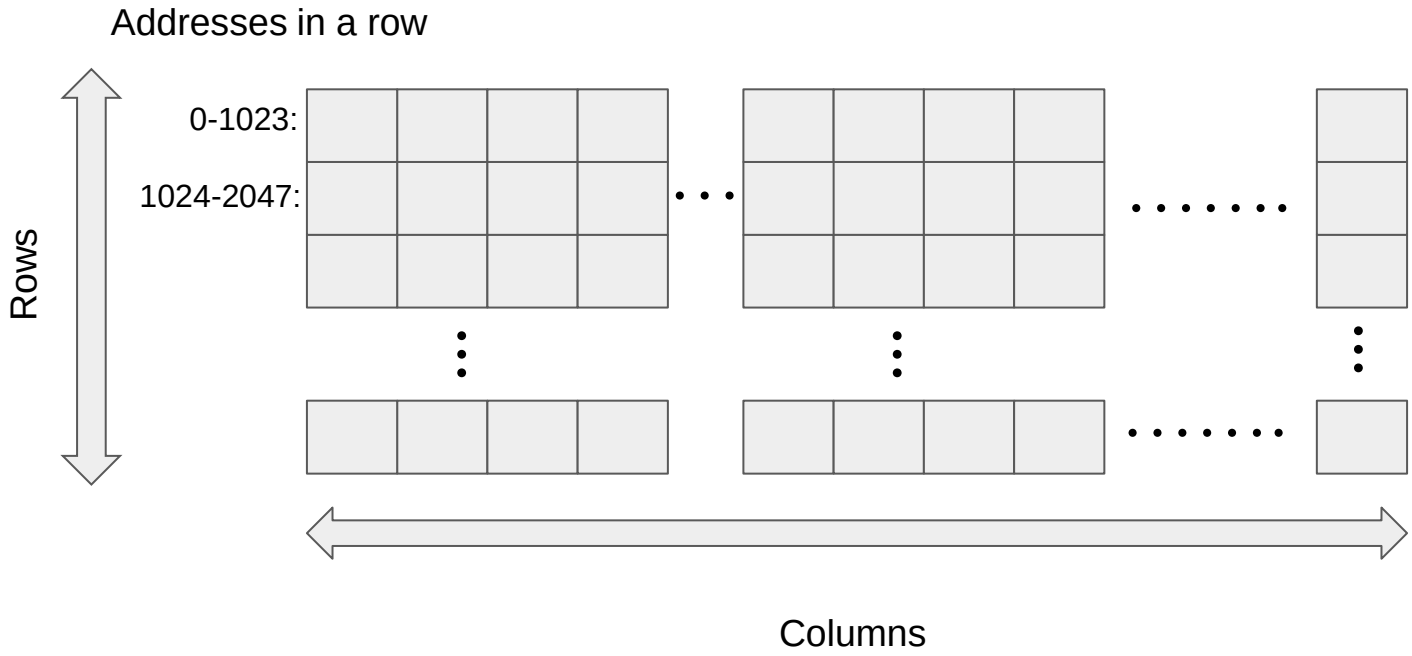
There are two test cases that differs only in the lw/sw implementation based on memory address format

Testcase1: accepts address stored in register

Testcase 2: accepts address as a constant

Assume ROW_ACCESS_DELAY=10 cycles, COL_ACCESS_DELAY =2 cycles

Test case description : DRAM implementation



Test case 1 description: cycle-wise

Cycle	Instruction	Registers	Memory	Remarks
1	addi $s0, $zero, 1000	s0=1000		
2	addi $s1, $zero, 1024	s1=1024		
3	addi $t0, $zero, 1	t0=1		
4	addi $t1, $zero, 2	t1=2		
5	sw $t0, 0($s0)			Instruction issues DRAM request, initially no row is active
6-15	DRAM: Activate row 0			Activate row 0 (1000 address lies in row 0)
16-17	DRAM column access and update		Address 1000=1	Column access takes 2 cycles and update the value
18	sw $t1, 0($s1)			DRAM request issued
19-28	DRAM: Writeback Row 0			As request is for 1024 address (row1), writeback activated row 0
29-38	DRAM: Activate row 1			Load row 1 in row buffer
39-40	DRAM: Column access		Address 1024=2	Column access and update

Test case 1 description: cycle-wise

Cycle	Instruction	Register	Memory	Remarks
41	lw $t2, 0($s0)			DRAM request issued
42-51	DRAM: Writeback row 1			As 1000 address value is to be loaded, writeback active row 1
52-61	DRAM: Activate row 0			Load row 0 in row buffer
62-63	DRAM: Column request	t2=1		Column access and write value in the register
64	lw $t3, 0($s1)			DRAM request issued
65-74	DRAM: writeback row 0			Writeback row 0, as request for 1024 address (row 1)
75-84	DRAM: Activate row 1			Activate row with address 1024
85-86	DRAM: Column access	t3=2		Column access and write the value in t3
87	add $t3, $t3, $t2	t3=1+2=3		
88	addi $s2, $zero, 1028	s2=1028		
89	sw $t3, 0($s2)			DRAM request to address 1028 (row 1) , currently active row is also 1
90-91	DRAM: Column access	-	Address 1028=3	As the row is already in row buffer, no activate/writeback time

Test case 2 description: cycle-wise

Note: In case of **test case 2** of part 1, reduce 3 cycles as instructions (addi) to load address (1000, 1024, 1028) to registers are not present.

Cycle	Instruction	Register	Memory	Remarks
1	addi $t0, $zero, 1	t0=1		
2	addi $t1, $zero, 2	t1=2		
3	sw $t0, 1000			Instruction issues DRAM request, initially no row is active
4-13	DRAM: Activate row 0			Activate row 0 (1000 address lies in row 0)
14-15	DRAM column access		Address 1000=1	Column access takes 2 cycles and update the value
16	sw $t1, 1024			DRAM request issued
17-26	DRAM: Writeback Row 0			As request is for 1024 address (row1), writeback activated row 0
27-36	DRAM: Activate row 1			Load row 1 in row buffer
37-38	DRAM: Column access		Address 1024=2	Column access and update
39	lw $t2, 1000			DRAM request issued
40-49	DRAM: Writeback row 1			As 1000 address value is to be loaded, writeback active row 1
50-59	DRAM: Active row 0			Load row 0 in row buffer

Test case 2 description: cycle-wise

Cycle	Instruction	Register	Memory	Remarks
60-61	DRAM: Column request	t2=1		Column access and write value in the register
62	lw $t3, 1024			DRAM request issued
63-72	DRAM: writeback row 0			Writeback row 0, as request for 1024 address (row 1)
73-82	DRAM: Activate row 1			Activate row with address 1024
83-84	DRAM: Column access	t3=2		Column access and write the value in t3
85	add $t3, $t3, $t2	t3=1+2=3		
86	sw $t3, 1028			DRAM request to address 1028 (row 1) , currently active row is also 1
87-88	DRAM: Column access		Address 1028=3	As the row is already in row buffer, activate and writeback time is avoided.

Part 2: Non-blocking memory

In the case of non-blocking memory, if there is a DRAM request issued then instead of waiting for its response to come back, processor starts executing instructions sequentially till it finds it to be safe (i.e., until the following instruction is not dependent on the response)

Let's consider a simple example with 3 instructions: ROW_ACCESS_DELAY=4, COL_ACCESS_DELAY=2

```
I1: lw $t0, 0($s0)
I2: addi $t1, $t1, 1                #independent of I1
I3: add $t0, $t0, $t1               #dependent on I1, as uses value loaded by lw to t0 register
```

Assumptions: $s0 stores memory address 1000 and have value 1, $t1=2

Cycle analysis:

Cycle 1: Execute I1, DRAM request issued but waiting for response (will take 6 cycles to come back)
Cycle 2: Execute I2 along with DRAM row_access for I1 (As I2 is independent, execute it)
Cycle 3-7: Only DRAM access is done and value is written to $t0. Reason being I3 is dependent on I1
Cycle 8: Execute I3