2.2 lw, sw: Load and store instructions

Load instruction: lw

A ***load instruction*** copies data from memory into a register. A MIPS load instruction format is shown below. Another section discusses the reason for the 0( ) around the memory-address.

```
lw register 0(memory-address)
```

MIPS register names start with a $. MIPSzy supports 8 registers. Writeable registers are $t0, $t1, ..., $t6. A special $zero register always has the value 0 and can only be read, not written.

The load instruction's memory-address is a register whose value is the memory address from which data is copied.

Load word

lw is short for "load word", in contrast to just loading a byte (a word is four bytes).

PARTICIPATION ACTIVITY 2.2.1: Load instruction: lw.

1) If $t6's value is 2020, what is the memory address being accessed by the following instruction?

```
lw $t0, 0($t6)
```

2020

Check Show answer

Correct

2020

The load instruction's memory address is the value in the register in the parentheses.

2) Given the following register file and memory contents, what value is loaded into

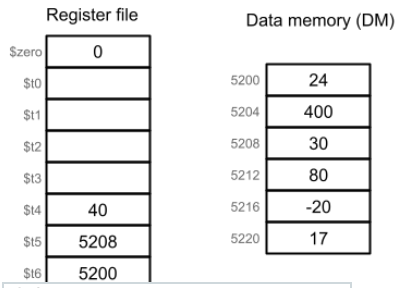
Correct

24

register $t3 by the following instruction?

The value in memory at address 5200 is 24. So, the load instruction writes 24 into register $t3.

```
lw $t3, 0($t6)
```



24

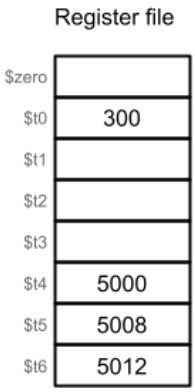
Check Show answer

3) Given the following register file, complete the load instruction to load register $t2 with data at memory address 5012.

Correct

$t6

$t6 holds 5012, so the instruction loads the memory value at address 5012 into $t2.




```
lw $t2, 0( $t6 )
```

Check Show answer

4) Assuming $t5 holds 6000, write a load instruction that loads register $t4 with data at memory address 6000.

Correct

lw $t4, 0($t5)

Loads value from memory address 6000 into register $t4.

lw $t4, 0($t5)

Check Show answer

Feedback?

Store instruction: sw

A ***store instruction*** copies data from a register to memory. A MIPS store instruction format is shown below. Another section discusses the reason for the 0( ) around the memory-address.

```
sw register 0(memory-address)
```

PARTICIPATION ACTIVITY 2.2.2: Store instruction: sw.

1) Assuming $t6 holds 600 and $t0 holds 5008, what is the memory address for the following instruction?

```
sw $t6, 0($t0)
```

5008

Check Show answer

Correct

5008

$t0's value is the memory address, and the $t6's value will be stored to memory.

2) Given $t2 holds 6200, $t3 holds 536, and $t4 holds 616, what value is stored into memory?

```
sw $t3, 0($t2)
```

536

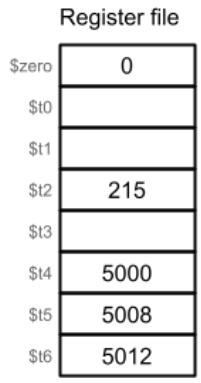
Check Show answer

Correct

536

$t3's value is 536. Thus, 536 is stored into memory at address 6200 ($t2's value).

3) Given the following register file, complete the store instruction to store register $t2's value into memory at address 5000.




```
sw   $t2   , 0($t4)
```

Check Show answer

Correct

$t2

$t2's value will be stored to memory at the address in $t4, which is 5000. So the instruction stores the value 215 to memory at address 5000.

4) Assuming $t0 holds 5400 and $t1 holds 280, write a store instruction that stores register $t1's value into memory at address 5400.

```
sw   $t1, 0($t0)
```

Check Show answer

Correct

sw $t1, 0($t0)

$t0 holds the memory address, which is 5400. So, the instruction stores $t1's value (280) into memory at address 5400.

Feedback?

Instruction format summary: lw, sw

The condensed instruction format below specifies all registers using $ followed by a single character. Ex: $a.

Table 2.2.1: Instruction summary: lw, sw.

Instruction	Format	Description	Example
lw	`lw $a, 0($b)`	Load word: Copies data from memory at address $b to	`lw $t3, 0($t6)`

		register $a.	
sw	`sw $a, 0($b)`	Store word: Copies data from register $a to memory at address $b.	`sw $t1, 0($t3)`

Feedback?

CHALLENGE ACTIVITY 2.2.1: Load and store instructions.

547404.4091098.qx3zqy7

Jump to level 1

Compute: DM[7612] = DM[7604]

```
lw  $t3 , 0( $t4 )
sw  $t3 , 0( $t5 )
```

Registers	
$t3	0
$t4	7604
$t5	7612

Data memory	
7604	9
7612	0

1 2 3

Check | Try again

Done. Click any level to practice more. Completion is preserved.

Your code with comments:

```
lw $t3, 0($t4)    # $t3 = M[$t4 + 0] = M[7604]

sw $t3, 0($t5)    # M[$t5 + 0] = M[7612] = $t3
```

Your code resulted in:

Registers	
$t3	9
$t4	7604
$t5	7612

Data memory	
7604	9
7612	9

Feedback?

How was this **Provide section feedback**

section?