5.5 Load and store with offsets

Load instruction with offset

An earlier section introduced the load instruction, which copies data from data memory into a register:

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

A memory address alone requires 32 bits, so cannot fit entirely within a 32-bit MIPS instruction. Thus, the memory address register.

Frequently, memory accesses are offsets from a base memory address, such as 5032 + 4, 5032 + 8, etc. Thus, the actual n is formed by adding a base memory address and an offset:

```
lw register offset(base-address)
```

Ex: If $t6 contains 5032, then `lw $t0, 4($t6)` copies the value in memory address 4 + 5032, or 5036, into $t0.

An ***offset*** is an amount added to a base address to form a final address. In MIPS, the offset is a 16-bit number so can rang 32,767.

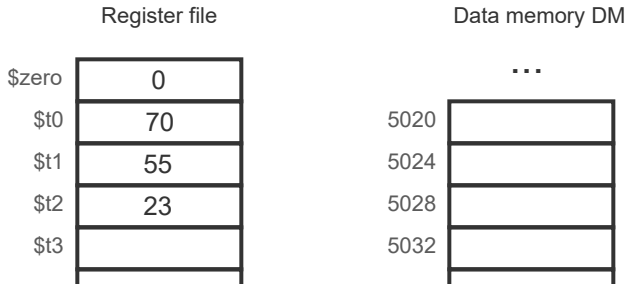
PARTICIPATION ACTIVITY 5.5.1: lw instruction with offset.

Start 2x speed

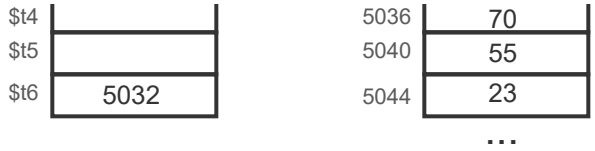
```
lw $t0, 4($t6)
```

4 + 5032 = 5036

$t0 = D(5036)



```
lw $t1, 8($t6)
lw $t2, 12($t6)
```



PARTICIPATION ACTIVITY 5.5.2: Load instruction.

1) Assuming $t6 holds 5032, what is the base address for:

```
lw $t5, 10($t6)
```

Check Show answer

2) Assuming $t6 holds 6044, what is the offset for:

```
lw $t4, 52($t6)
```

Check Show answer

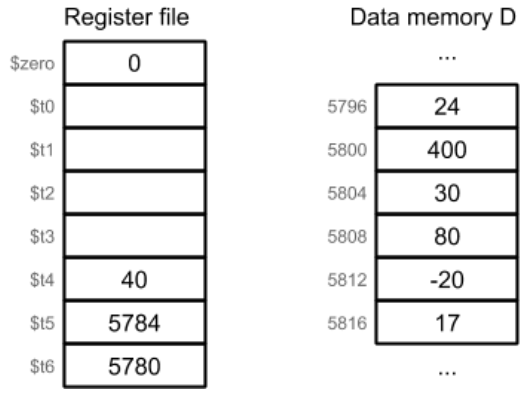
3) Assuming $t6 holds 5072, from what memory address is the value loaded for:

```
lw $t5, 24($t6)
```

Check Show answer

4) Given the following register file and memory contents, what value is loaded into register $t3 by:

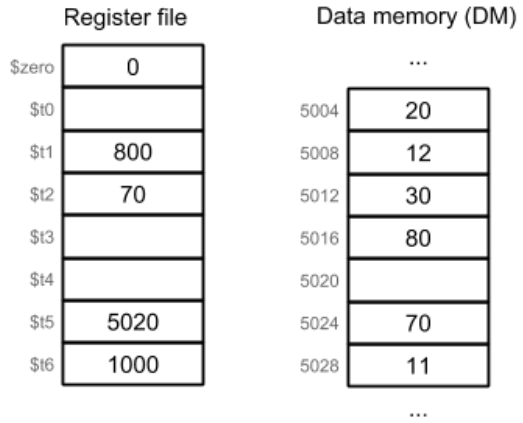
```
lw $t3, 20($t6)
```



Check Show answer

5) Given the following register file and memory contents, what value is loaded into register $t3 by:

```
lw $t3, -4($t5)
```



Check Show answer

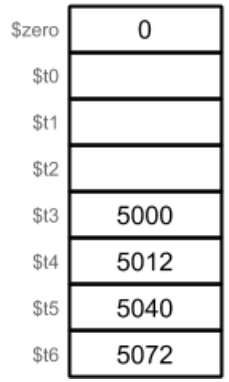
6) Assuming $t6 holds 5020, complete the load instruction to load register $t2 with data at memory location 5044 using $t6 as the base address.

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

Check Show answer

7) Given the following register file, complete the load instruction to load register $t2 with data at memory location 5040, using an offset of 28.

Register file




```
lw $t2, 28(    )
```

Check Show answer

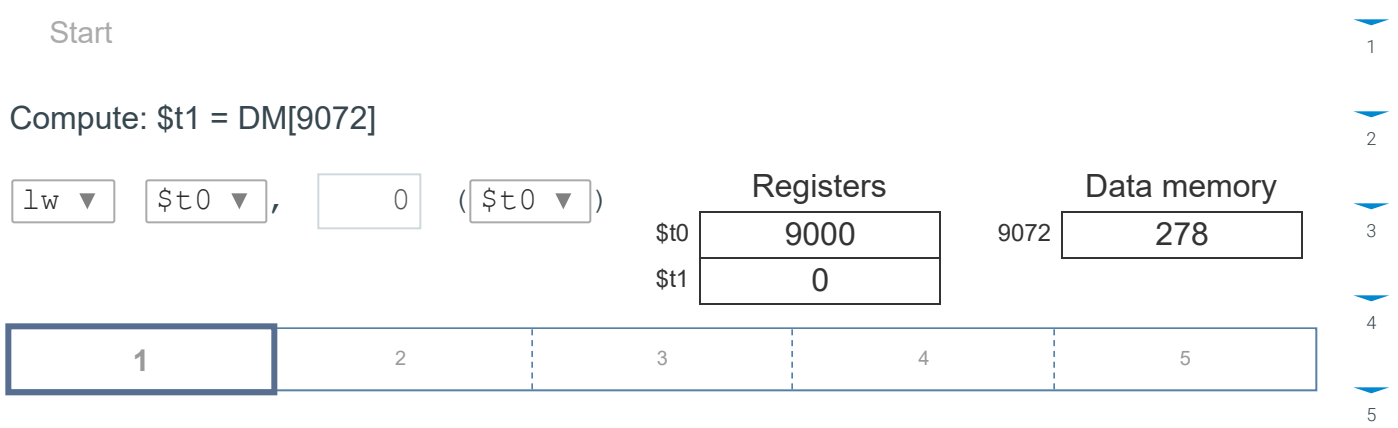
8) Assuming $t5 holds 5000, write a load instruction that loads register $t4 with data at memory location 5048, using $t5 as the base address.

Check Show answer

9) Assuming $t5 holds 6000, write a load instruction that loads register $t3 with data at memory location 5960, using $t5 as the base address.

Check Show answer

CHALLENGE ACTIVITY 5.5.1: Loading and storing from memory.



Store with offset

An earlier section introduces a store instruction, which copies data from a register to memory. As with a load instruction, th address is formed by adding a base-address plus an offset.

```
sw register offset(base-address)
```

PARTICIPATION ACTIVITY 5.5.3: Store instruction with offset.

1) Assuming $t3 holds 5132, which store instruction stores the value of register $t2 to data at memory location 5144 using $t3 as the base address?

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

```
sw $t2, $t3(12)
```

- ○ `sw $t2, 12($t3)`

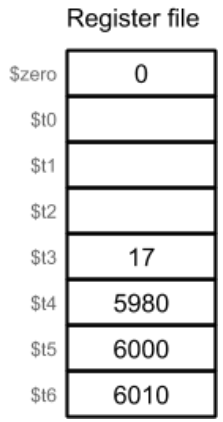
2) Given the following register file, which instruction stores register $t3 to memory location 5084?

Register file

$zero	0
$t0	
$t1	
$t2	
$t3	17
$t4	5084
$t5	5076
$t6	5080

- ○ `sw $t3, 20($t4)`
- ○ `sw $t3, 8($t5)`
- ○ `sw $t3, 8($t6)`

3) Given the following register file, which instruction stores register $t3 to memory location 5984?



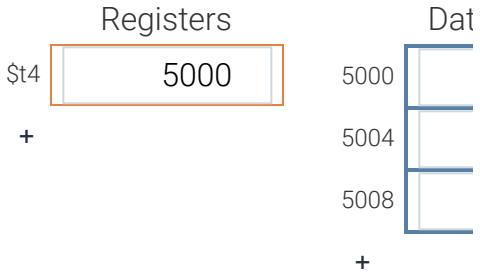
- `sw $t3, -4($t4)`
- `sw $t3, -16($t5)`
- `sw $t3, -8($t6)`

PARTICIPATION ACTIVITY 5.5.4: Load, store, and memory.

1. Run the simulation step-by-step, observing register and memory values.
2. Load DM[5008]'s value into register $t3 using register $t4 as the base address and an offset of 8.
3. Add DM[5008]'s value to register $t1's value, which already holds the sum of DM[5000] and DM[5004].
4. Store the addition result in DM[5012].

Assembly

```
Line 1  lw $t1, 0($t4)    # Load DM[5000]
Line 2  lw $t2, 4($t4)    # Load DM[5004]
Line 3  add $t1, $t1, $t2 # Add DM[5000] and DM[5004]
```



ENTER SIMULATION STEP RUN

More options

Instruction format summary: lw and sw with offsets

In the condensed instruction format below, `C` is a literal value, like 20 or -4.

Table 5.5.1: Instruction summary: lw and sw with offset.

Instruction	Format	Description	Example
lw	`lw $a, C($b)`	Load word: Copies data from memory at address $b + C to register $a.	`lw $t3, 20($t6)`
sw	`sw $a, C($b)`	Store word: Copies data from register $a	`sw $t1, -4($t3)`

		to memory at address $b + C.	

Provide feedback on this section