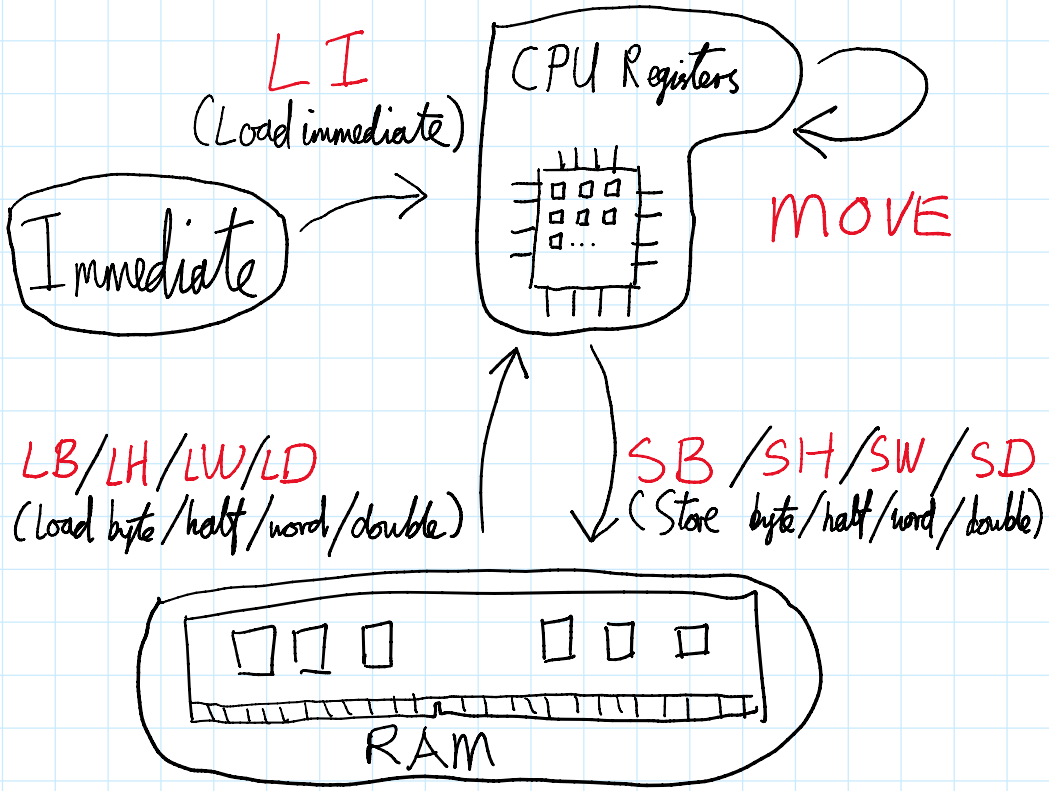
Instructions to Move Stuff Around

Thursday, 16 June 2022 5:53 PM



LI

Loads an immediate (a number, or a character) into a register. The following loads the number 42 into register t0.

```
li      $t0, 42
```

MOVE

Moves values from a register to another register. The following moves (copies) whatever's in register t0 to register t1.

```
move    $t1, $t0
```

LB/LH/LW/LD

Loads a byte, half word (2 bytes), word (4 bytes), double word (8 bytes) from somewhere in memory. The tricky part is specifying where in memory. In MIPSY, there are 6 ways (as per the MIPS documentation) to do so:

Format	Example	Meaning
`Label`	`lw $t0, my_label`	Load 4 bytes from 'my_label' and put it in register t0.
`(Rn)`	`lw $t0, ($t1)`	If you have an address (aka pointer) stored in register t1, perhaps from an earlier LA, load 4 byte from whatever t1 is pointing to. Assuming t1 holds 0x10010000, this instruction loads 4 bytes starting from 0x10010000.
`Imm(Rn)`	`lw $t0, 16($t1)`	Same as above, except we add 16 to whatever's in register t1. Assuming t1 again holds 0x10010000, this instruction loads 4 bytes starting from 0x10010010.
`Label(Rn)`	`lw $t0, my_label($t1)`	Add whatever's in register t1 to the address of 'my_label', and load 4 bytes from there. Assuming 'my_label' is at 0x10010000, and t1 holds 4, this instruction loads 4 bytes starting from 0x10010004.
`Label ± Imm`	`lw $t0, my_label + 16`	Add 16 to the address of 'my_label', and load 4 bytes from there. Assuming 'my_label' is at 0x10010000, this instruction loads 4 bytes starting from 0x10010010.
`Label ± Imm(Rn)`	`lw $t0, my_label + 16($t1)`	Adds 16, and whatever's in t1, to the address of 'my_label', and load 4 bytes from there. Assuming 'my_label' is at 0x10010000, and t1 holds 4, this instruction loads 4 bytes starting from 0x10010014.

Note that using 'LD' puts the lower word in the specified destination register, and the upper word in the next register. For example:

```
ld      $t0, my_label
```

will spread the 8 bytes at 'my_label' across register t0 and t1.

SB/SH/SW/SD

Stores a byte, half word (2 bytes), word (4 bytes), double word (8 bytes) from a register to some address in memory. The addressing modes are exactly the same as the above. Note this is one of the few instructions where the operand ordering is the opposite to what you'd do in C. The below stores 4 bytes from register t0 to 'my_label' in memory.

```
sw      $t0, my_label
```

LA (Not Pictured)

Loads the address of a label into a register. The following loads the address of 'my_label' into register t0.

```
la      $t0, my_label
```