Memory Organisation

Slide 1

- Memory
- Reading and writing data from memory
- Arrays
- Strings

Slide 2

- The memory of the computer holds both the program and data.
- We only have 32 registers avaliable, even in small programs you are going to run out of space to store data.
- We can use memory to store data.

Slide 3

Memory is organised as a sequence of bytes:

Address	Value
0	8 bit Value
1	8 bit Value
2	8 bit Value
⋮	⋮

Every value in memory has an address, the memory is continuous every element can be accessed in the same way.

Slide 4

- Remember registers hold 32 bits, that is 4 bytes (a word).
- You spend a lot of time reading and writing registers.

This means you often have to think of memory in chunks of 4.

Address	Value
0	32 bit Value
4	32 bit Value
8	32 bit Value
12	32 bit Value
⋮	⋮

- To read information from memory you us the, `lw`, load word instruction.

Assume `$s0` holds the address `0x8000000` then

```
lw $t0,0($s0)
```

Slide 5

Will load the contents of memory location `0x8000000` into `$t0`

and

```
lw $t0,4($s0)
```

will load the contents of memory location `0x8000004` into `$t0`.

Format of `lw`.

Slide 6

- `lw register,constant(register)`

The constant can not be a register.

Slide 7

How do we load an address into a register? We can you the same trick as in the previous lecture, but there is a pseudo instruction:

- `la $t0,address`

There is a reason that you you `la` rather than `li`, but I can't tell you what it is yet.

When you start doing your labs you'll start to learn how to use labels.

Slide 8

To store a value from a register into a memory location you use, `sw`, store word. This instruction has the same format as `lw`.

```
la $s0,0x8000000
li $t0,10
sw $t0,0($s0)
sw $t0,4($s0)
```

This puts the value 10 into locations `0x8000000` and `0x8000004`.

Slide 9

- To access the `i`th element of an integer array you need to access the memory address

$$\text{Base\_Address} + 4 * i$$

You must remember to do this.

Slide 10

- Sometimes, especially when you are dealing strings you want to read and write bytes.
- The MIPS processor has two instructions `lb` and `sb` which read and write bytes, these have the same format as `lw` and `sw`.

Revision

Slide 11

- `la` to load an address into a register. Remember there is a distinction between the address and the value stored at that and `sw`address. (Pointers and values).
- `lw` load a value into a register from memory, `sw` store a value and `sw` from a register to memory.
- Integer arrays, multiply by 4.
- Strings and byte arrays, use `lb` and `sb`.