Mips Registers

Pre Lecture Video Notes

An instruction is a command that hardware understands

- Instruction set is the vocabulary of commands understood by a given computer
- Included arithmetic instructions, memory access instructions, logical operations, instructions for making decisions

The general classes of MIPS instructions are

- Arithmetic
  - add, subtract, multiply, divide
- Logical
  - and, or, not, not shift
- Data transfer
  - load from or store to memory
- Transfer control
  - Jumps, branches, calls, returns

Arithmetic Instructions

- Each MIPS only performs on operation
- Each one must have 3 variables
- These variables can be the same

```
add a, b, c     # a = b + c;
add a, a, a.    # a = a + a

sub a, b, a   # f contains t0 - t1

# a = b - ((b+c) + a + a)
```

- if the statement is more complex we need to break it into pieces

In MIPS, operands for general arithmetic operations must be from registers of constants

- 32 programmer useable registers

- Reflects *"Faster is better"*
  Registers use less power

Name	Number	Use
`$zero`	0	Const 0
`$at`	1	Assembler temporary, for resolving pseudoinstructions
`$v0 – $v1`	2-3	Function results and expression evaluation
`$a0 – $a3`	4-7	Arguments
`$t0 – $t9`	8-15, 24-25	Temporary
`$s0 – $s7`	16-23	Saved Temporary
`$k0 – $k1`	26-27	OS Kernel
`$gp`	28	Global pointer
`$sp`	29	Stack pointer
`$fp`	30	Frame pointer
`$ra`	31	Return address

More about how these work: Intro To MIPS
More about the use of these: Logical Operations

The general form for a mips instruction is

```
Instruction_mnemonic $target, $source1, $source2
```

- MIPS is case insensitive (not case sensitive) so `Add` can be `ADD` or even `aDd`

We don’t have variables, we have registers.