"Open the pod-bay doors, HAL."
Computer instruction from *2001: A Space Odyssey*

Machine Instructions

Computers do exactly what we tell them to do. Instructions are the medium for all tasks -- arithmetic, logic, I/O. Though most ISAs have a unique instruction set, the similarities are greater than the differences, justifying our study of the MIPS ISA alone.

Binary Review

Decimal notation represents numbers by their coefficients of powers of 10. Binary expresses numbers using coefficients of powers of 2.

321_{10}
$= 3 \times 10^2 + 2 \times 10^1 + 1 \times 10^0$
$= 300 + 20 + 1$
$= 256 + 64 + 1$
$= 1 \times 2^8 + 0 \times 2^7 + 1 \times 2^6 + 0 \times 2^5 + 0 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$
$= 101000001$

Binary Facts

- Alternate forms of binary:
  - Octal: symbols 0-7 and is convertable from binary in groups of 3
  - Hexadecimal: 0-9, A,B,C,D,E,F and is convertable from binary in groups of 4

$101000001_2 = 101\ 000\ 001 = 501_8$
$= 1\ 0100\ 0001 = 141_{16}$

1011 1010 1101 = BAD

Fast Approximations
$2^{10} \approx 1{,}000$
$2^{20} \approx 1{,}000{,}000$
$2^{30} \approx 1{,}000{,}000{,}000$
$2^{40} \approx 1{,}000{,}000{,}000{,}000$

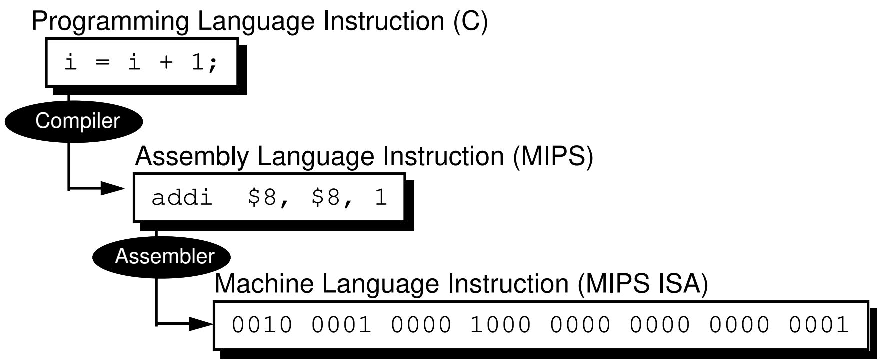
Instruction Interpretation

- A computer is a hardware interpretation of the "fetch/execute" cycle

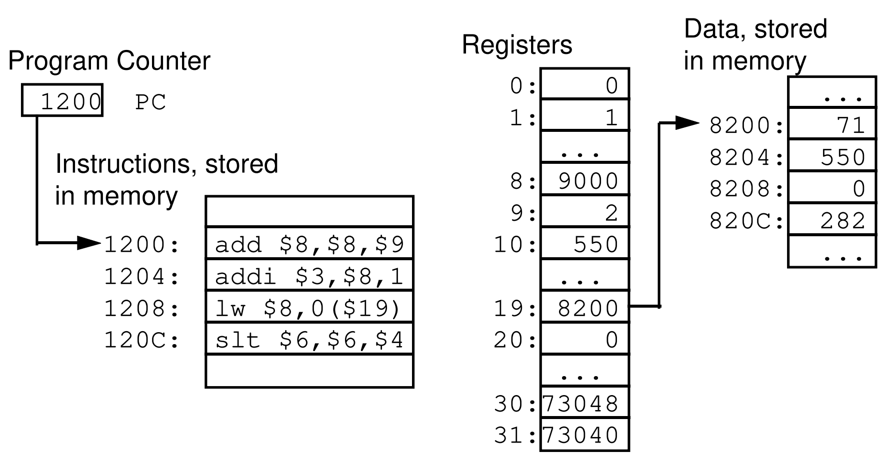
```
fetch an instruction from memory[PC]
decode the instruction
access the operands of the instruction
perform (execute) the operation on the operands
      placing the results in a register
PC  :=  PC+1
```

Translating Instructions

- The instructions must have a form understandable to the follower. Computers only understand binary, so translations are needed to lower the level.



Components of Instruction Execution



Anatomy of an Instruction

Operator and operands

- Operation -- assembly language uses prefix
- Operands separated by commas
- Comments follow #

```
add   $8, $9, $10 # Reg8 = Reg9 + Reg10
```

- ISAs usually use a few instruction formats, each format customized to a particular type of instruction

R type Instruction Format

- “R” Type Instructions have all operands in registers
    - There are exactly 3 operands
    - Operands can only be register names
    - Values of operands are integers, with the default being signed integers

```
Reg7 = (Reg8-Reg9)+(Reg6-Reg5
becomes
     sub $10, $8, $9
     sub $11, $6, $5
     add $7, $10, $11
```

Encoding R-type Instructions

- All MIPS instructions are 1 word (32 bits) long
- Partition word into fixed length fields
    - op = operation
    - rd = destination operand
    - rt = second source operand
    - rs = first source operand
    - shamt = shift amount
    - funct = function field

```
add  $8,  $9  $10  # no comment
```

op	rs	rt	rd	shamt	funct
0	9	10	8	0	32
000000	01001	01010	01000	00000	100000
← 6 →	← 5 →	← 5 →	← 5 →	← 5 →	← 6 →