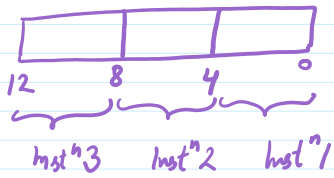
Lecture-6 (RISC and CISC)

29 January 2024 14:56

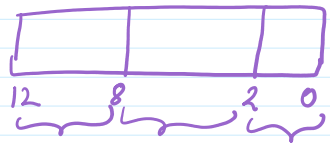
Encoding of instructions :-

① Fixed-length instruction encoding.



[Every instruction is represented using same number of bytes.]

② Variable-length instruction encoding.



[different computer instructions is represented using different number of bytes.]

Different real-life processors :-

① IBM 360/370 (1960-70)

② VAX-11/780 (1970-80)
by DEC (Digital Equipment Corporation)

③ Intex x86 / Pentium (1985- present)

④ MIPS (1980-90)

MIPS Technologies (US)

(MIPS32 : A case study)

⑤ SPARC (Scalable Processor ARChitecture)

by Sun Microsystems

⑥ ARM Microcontroller

Apple A16 Bionic
(ARM-based)

Google pixel →
Qualcomm Snapdragon 821
(ARM-based)

Two broad classification of instruction sets

① Complex Instruction Set Computer [CISC]

② Reduced Instruction Set Computer [RISC]

Set Computer [CISC]

Set Computer [RISC]

→ Instruction sets are complex

→ Large number of addressing modes (R-R, R-M, M,M, ..., etc.)

→ Special-purpose registers and flags.

→ Variable - length instructions / Complext instruction encoding.

→ Instruction decoding more complex

→ Control unit design complex

→ Complex pipeline implementation.

→ CISC Examples

① IBM 360/370

② VAX-11/780

③ Intel x86/Pentium

Only CISC present today Because translation CISC → RISC

possible.

② Reduced Instruction Set Architecture (RISC)

→ Widely used today.

→ LOAD - STORE Architecture.

Only LOAD and STORE instructions access memory.

All other instructions operate on processor registers.

→ Efficient Pipelining
simple architecture.

→ Fixed - length instruction Encoding.

→ RISC examples

MIPS Family
SPARC
ARM

CISC ↔ RISC Conversion :-

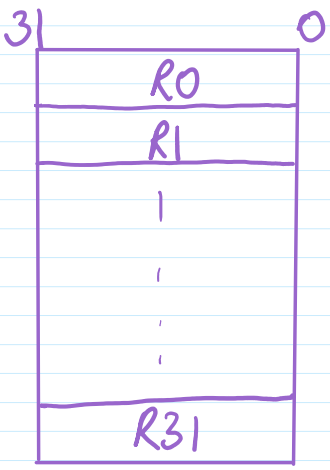
CISC → RISC

```
SUB  r4, (r1), (r2)
MUL  r7, r4, r6
ST   (r6), r7
```

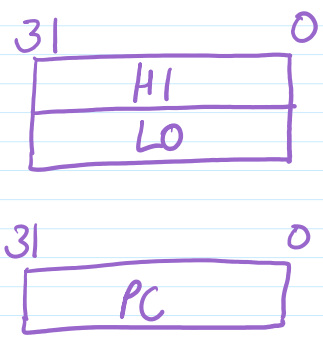
```
LOAD r8, (r1)
LOAD r9, (r2)
SUB  r4, r8, r9
MUL  r7, r4, r6
ST   (r6), r7
```

CASE STUDY : MIPS32 Architecture

① MIPS32 Registers



General Purpose Registers



Special Purpose Registers

Eg :- Division

- Quotient (in LO)
- Remained (in HO)

② MIPS32 Instruction Set (32 bits)

Groups

- Load and Store
- Arithmetic and Logical

→ Jump and Branch
→ Miscellaneous (Conditional MOVE, NOP)
→ Coprocessor instruction (CP0, CP1, CP2, CP3)
Virtual Memory / Exception Handling / Reserved for future.

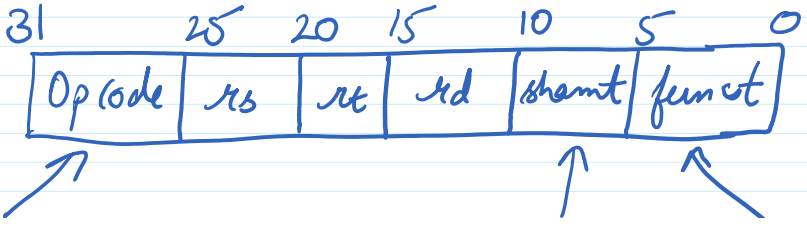
③ MIPS 32 Programming

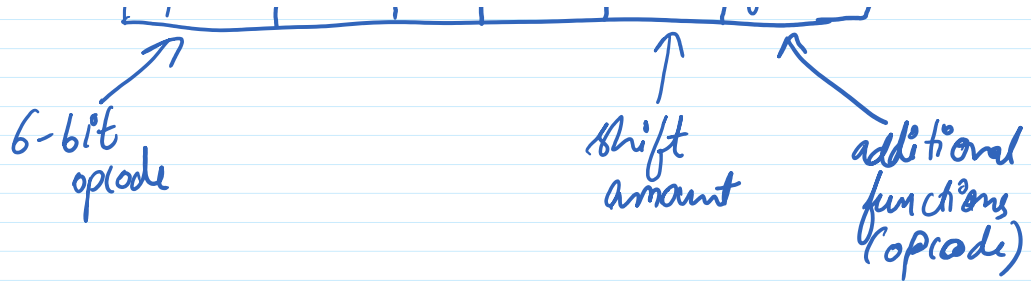
C Code	MIPS 32 Code
`A = B+C;` →	`add $s1, $s2, $s3`

`A= B+C-D;`
`E=F+A;` →
	`add $t0, $s1, $s2`
	`sub $s0, $t0, $s3`
	`add $s4, $s5, $s0`

④ MIPS 32 Instruction Encoding

→ R-type (Register)

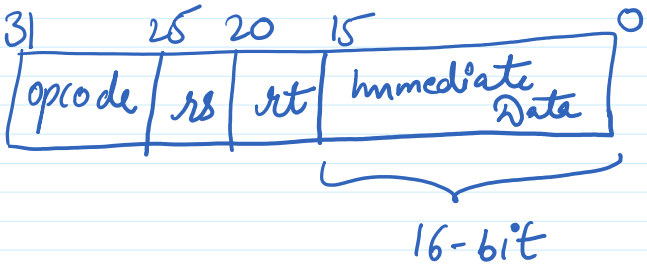




add $s1, $s2, $s3
sub $t1, $s3, $s4
sla $s1, $s2, 5

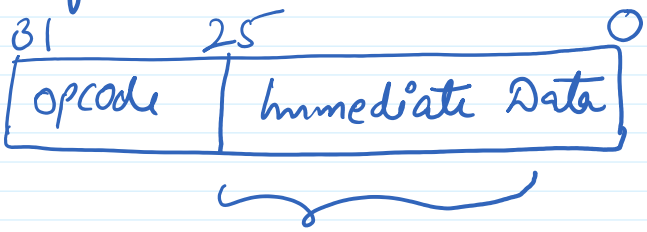
→ I-type (Immediate)

Contains 16-bit immediate data field.



addi $t0, $s1, 188

→ J-type (Jump)



j Label

j Label

⑤ Addressing Modes in MIPS 32

→ Register addressing
→ Immediate addressing
→ Index Addressing
→ Relative Addressing (16-bit offset + PC)