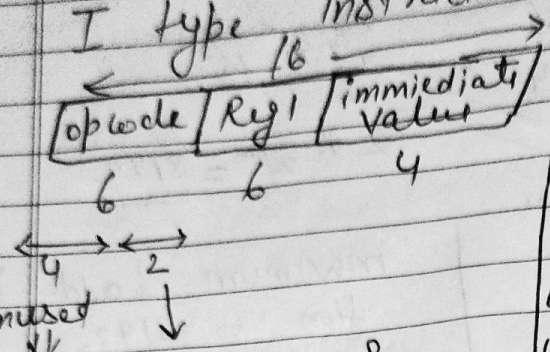
64 register → $\log_2 64$ → 6 bit size for reg

16 bit instruction

I type instruction



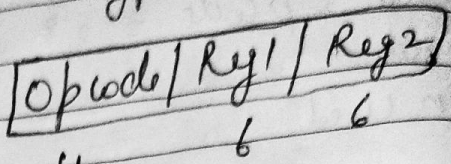
$(16-x) * 2^2 = 8$

$(16-x) \times 4 = 8$

$16 - x = \frac{8}{4} 2$

$14 = x$ Ans

R-type instruction



max opcodes $= 2^4 = 16$

used " $= x$

unused " $= 16 - x$

---

* How instructions are generated Lecture 9 and executed?

- Compiler generates instructions based on CPU.

⇒ means only those type of instructions it will generate, what are are supported by CPU.

MY CHOICE
Date:
Page No.:

Gate question (2006-07) Not sure

In a simplified computer the instructions are:

OP R_j, R_i → Perform R_j OP R_i and stores the result in register R_i.

OP m, R_i → Perform val OP R_i and stores the result in R_i. Val denotes the content of memory location m.

MOV m, R_i → Moves the content of memory location m to register R_i.

MOV R_i, m → Moves the content of register R_i, to memory location m.

The computer has only two registers, and OP is either ADD or SUB. Consider the following basic block:

$t_1 = a + b$
$t_2 = c + d$
$t_3 = e - t_2$
$t_4 = t_1 - t_3$

Min No. of MOV instruction in the generated code

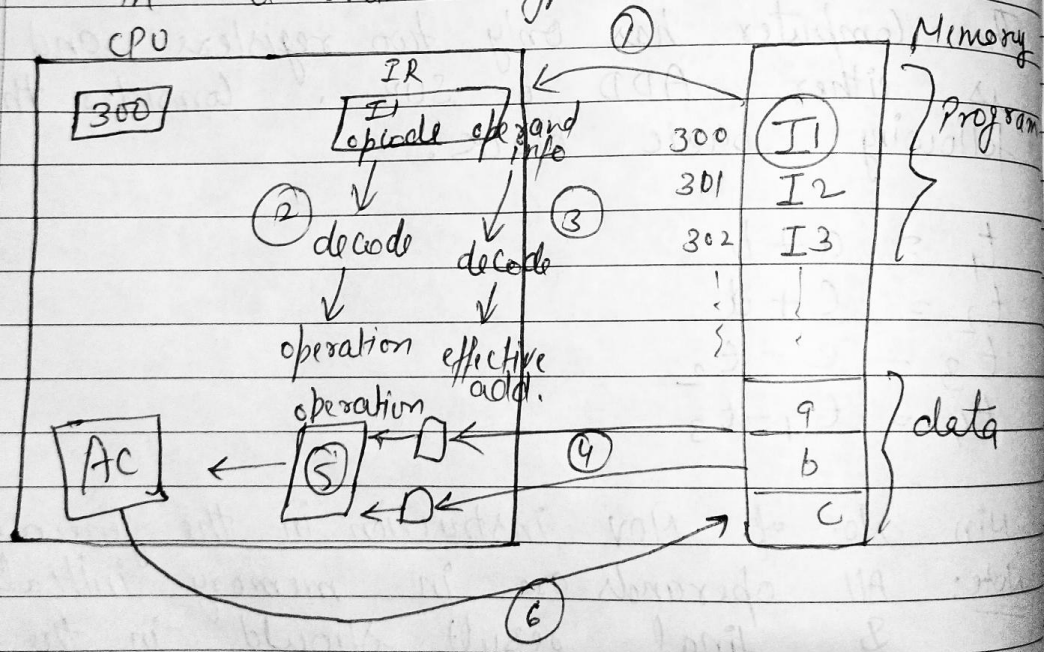
Note: All operands are in memory initially & final result should in the memory

$t_1 = a+b$	MOV b, R1	R1 ← M[b]
$t_2 = c+d$	ADD a, R1	R1 ← M[a] + R1
$t_3 = e-t_2$	MOV d, R2	R2 ← m[d]
$t_4 = t_1 t_3$	ADD c, R2	R2 ← m[c] + R2
	SUB e, R2	R2 ← m[e] - R2
	SUB R1, R2	R2 ← R1 - R2
	MOV R2, x	m[x] ← R2

Ans = 3, minimum 3 MOV used

* How instructions are executed?

o Effective address
→ Address of operand in a computation-type instruction or the target address in a branch-type instruction.



Instruction cycle

1) Instruction fetch
2) Instruction decode
3) Effective Add. Calculation
4) operand fetch
5) Execution
6) write back result

* Fetch and execution cycle
  - Fetch cycle
    - Instruction fetch

  Rest are execution cycle

Note : every type of instruction will not req. all the 6 phases but, instruction fetch, Instruction decode and execution mandatory.