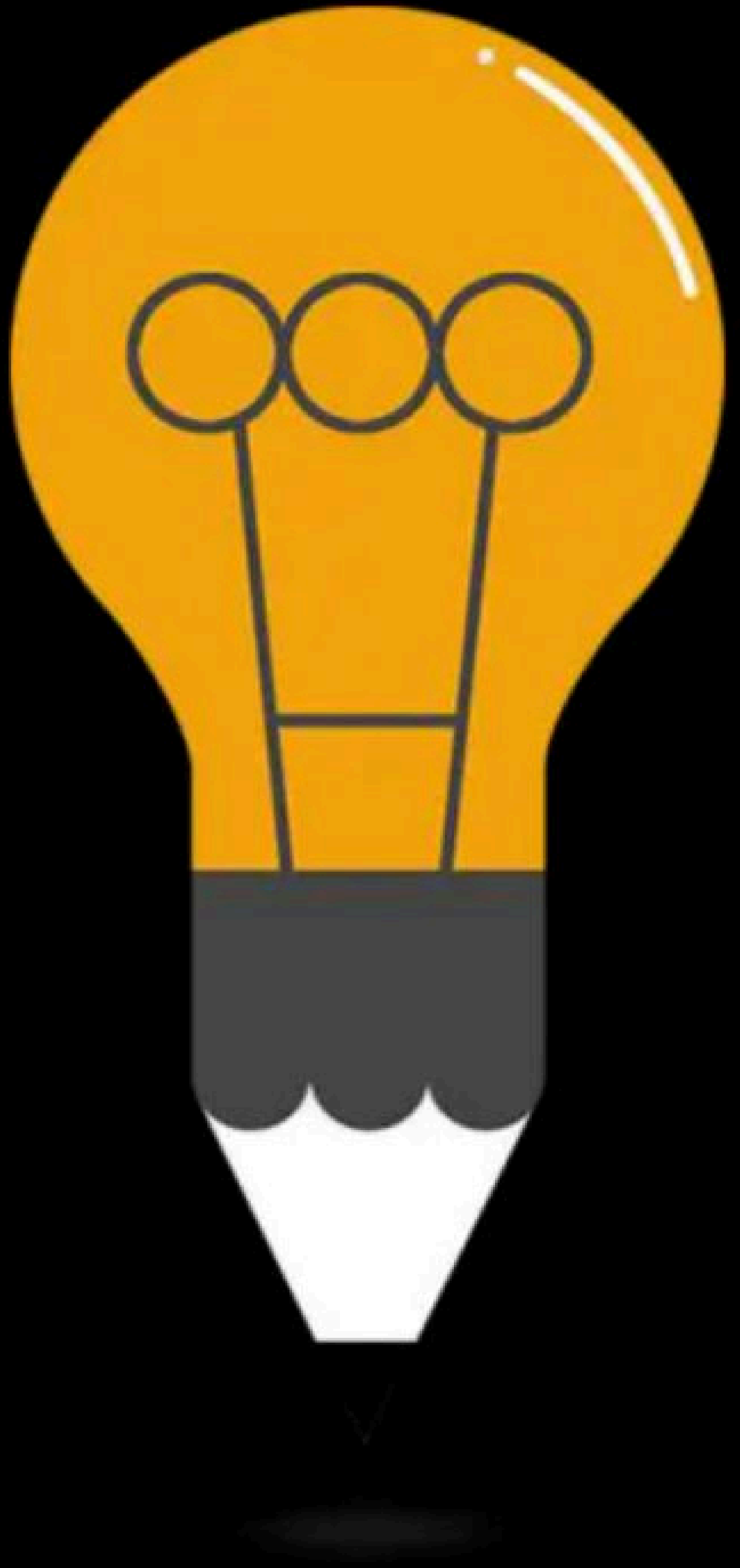
Instruction: Part III

Complete Course on Computer Organization & Architecture for GATE 2024 & 2025

Vishvadeep Gothi • Lesson 10 • Feb 23, 2023



Doubts & Instruction Practice & Instruction Cycle

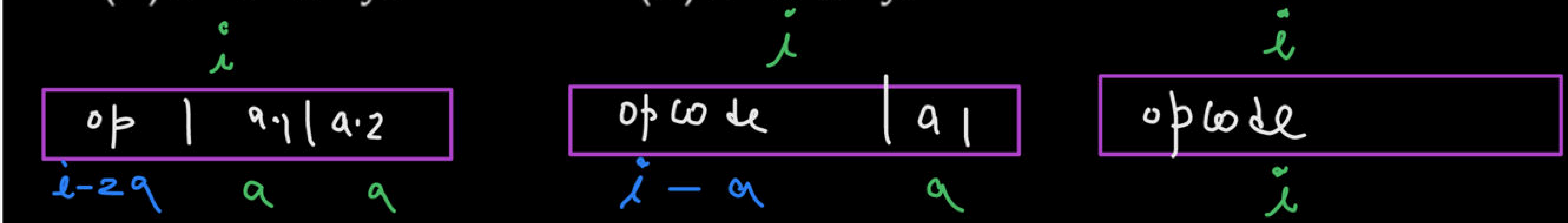
By: Vishvadeep Gothi

Question

Consider a system which supports 2-address, 1-address and 0-address instructions. The system has ‘i’ bits instructions and ‘a’ bits addresses. If there are ‘x’ 2-address instructions and ‘y’ 1-address instructions then which of the following is correct for maximum number of 0-address instructions supported by system?

(A) $2^{i} - 2^{a}x - y$ (B) $2^{i} - 2^{2a}x - y$

(C) $2^{i} - 2^{2a}x - y2^{a}$ (D) $2^{i} - 2^{a}x - y2^{a}$



$$x * 2^{2a} + y * 2^{a} + z = 2^{i}$$

$$z = 2^{i} - x2^{2a} - y2^{a}$$

instⁿ ⇒ i bits

add. ⇒ a bits

$\underset{x}{\text{3-add}}$, $\underset{y}{\text{2-add}}$, $\underset{z}{\text{1-add}}$. supported

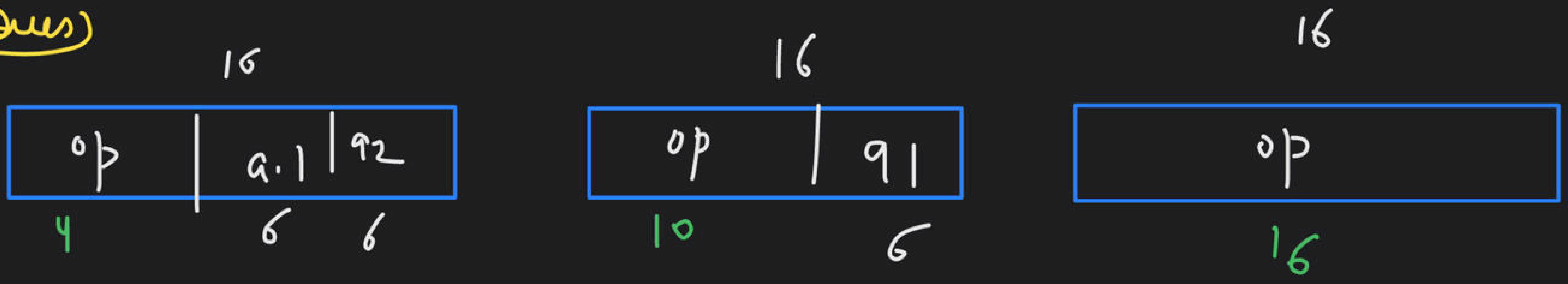
$$2^i = x * 2^{3a} + y * 2^{2a} + z * 2^a$$

op	a1	a2	a3
	a	a	a

op	a1	a2
	a	a

op	a.1
	a

Ques)



15 2-add inst^{ns}

60 1 add inst^{rs}

max ⇒ 0-add inst^{n}

$$15 * 2^{12} + 60 * 2^{6} + x = 2^{16}$$

$$\boxed{x = 256}$$

Question

Consider a system which supports 2-address, 1-address and 0-address instructions. The system has ‘a’ bits instructions and supports 2^m bytes memory. If there are ‘t’ 2-address instructions and ‘ w’ 1-address and ‘z’0-address instructions, then which of the following expression is correct?

add. = m - bits

(A) $2^a = 2^m t + w + z$

[✓ (B)] $2^a = 2^{2m} t + 2^m w + z$

(C) $2^a = 2^{3m} t + 2^{2m} w + 2^m z$

(D) $2^a = 2^{2m} t + 2^m w - z$

$$2^a = t * 2^{2m} + w * 2^m + z$$

ans = 192

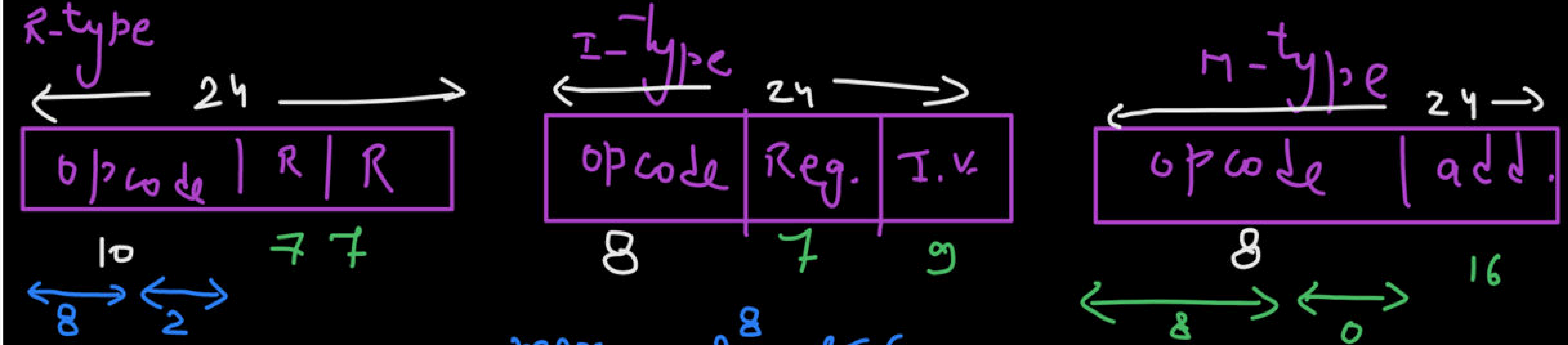
Question

Reg. = 7 bits

add. = 16-bits

24-bits

A CPU has 128 registers, 64KB byte addressable memory and 3Bytes instructions. The CPU supports 3 types of instructions: R-type, I-type and M-type. Each R-type instruction contains an opcode and 2 register names. Each I-type Instruction contains an opcode, a register name and a 9-bit immediate value. Each M-type instruction contains an opcode and a memory address. If there are 85 M-type instructions, 123 I-type instructions then maximum how many R-type instructions the CPU can support?



$48 * 2^2 = 192$

max $= 2^8 = 256$

used $= 123$

unused $= 133$

$133 * 2^0 = 133$

used $= 85$

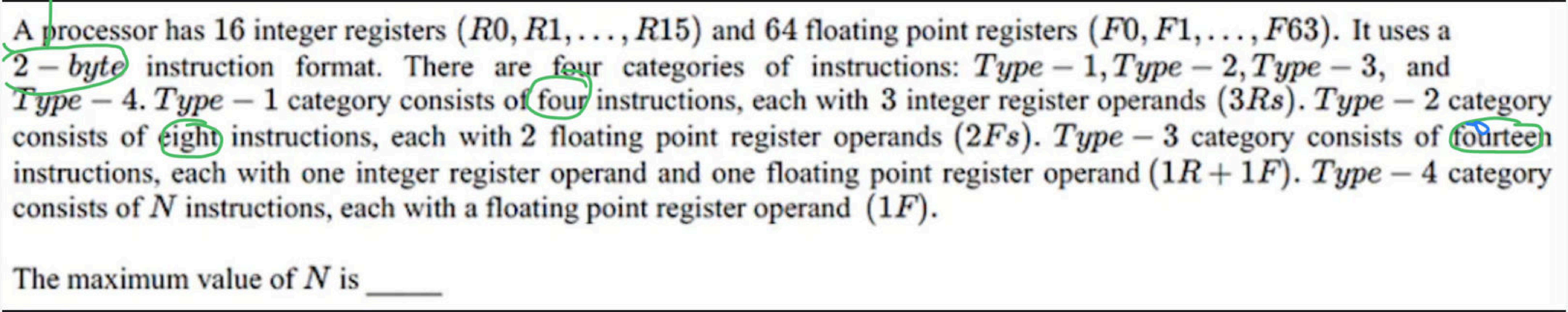
unused $= 48$

GATE - 2018

16-bits

R => 4 bits

F => 6-bits



A processor has 16 integer registers $(R0, R1, \ldots, R15)$ and 64 floating point registers $(F0, F1, \ldots, F63)$. It uses a $2-byte$ instruction format. There are four categories of instructions: $Type-1, Type-2, Type-3,$ and $Type-4$. $Type-1$ category consists of four instructions, each with 3 integer register operands $(3Rs)$. $Type-2$ category consists of eight instructions, each with 2 floating point register operands $(2Fs)$. $Type-3$ category consists of fourteen instructions, each with one integer register operand and one floating point register operand $(1R+1F)$. $Type-4$ category consists of N instructions, each with a floating point register operand $(1F)$.

The maximum value of N is _____

Ans = 32

Type 1

16

op	R	R	R
4	4	4	4

max = 2^4 = 16

used = 4

unused = 12

Type 2

16

op.	F	F
4	6	6

4 0

$12 * 2^6 = 12$

used = 8

unused = 4

T-3

16

op.	R	F
6	4	6

4 2

$4 * 2^2 = 16$

used = 14

unused = 2

T-4

16

opcode	F
10	6

6 4

$2 * 2^4 = 32$ Ans

Question

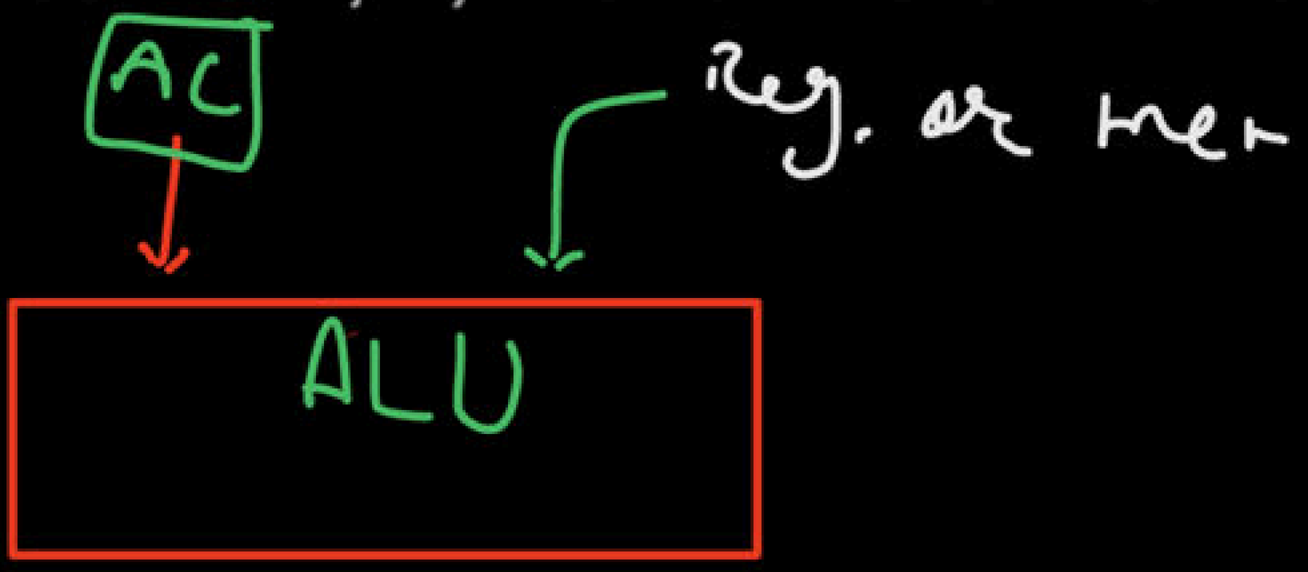
Ans = $\underline{\underline{2}}$

Consider an AC-based architecture system. For this system the following intermediate code is going to be converted in machine code. Minimum how many registers are required in system so that the code can run without register spill?

Consider second operand can be a register or memory operand

```
t1 = X + Y
t2 = t1 – Z
t3 = t1 + t2
t4 = t3 + M
```

Assume X, Y, Z and M are memory operands



AC ← X

AC ← AC + Y

R1 ← AC

AC ← AC – Z

R2 ← AC

AC ← R1

AC ← AC + R2

AC ← AC + M

mem. ← AC

Question

Consider there are 4 types of instructions in system:

1. Type 1: One opcode and 2 registers
2. Type 2: One opcode and 1 register
3. Type 3: One opcode and 1 memory address
4. Type 4: One opcode, 1 register and 1 memory address

Number of registers = 128 ⇒ Reg. = 7-bits

Maximum instruction length: 32bits (Variable length instructions)

→ fixed length opcode

Total Instructions:

Type-1: 15, Type-2: 20, Type-3: 12, Type-4: 14

→ Total = 61

⇓

opcode = 6 bits

Memory address size = ______ bits

Type 1:-

opcode	R.1	R.2
6	7	7

=> 20 bits

Type 2:-

opcode	Reg
6	7

=> 13 bits

Type 3:-

opcode	mem. add.
6	19

=> 25 bits

Type 4:-

← 32-bits →

opcode	Reg.	mem_add
6	7	19

Ans = 19

Ans = 3

Question

Consider a register-based architecture system which can support maximum 2-address instructions. For this system the following intermediate code is going to be converted in machine code. Minimum how many registers are required in system so that the code can run without register spill?

t1 = X + Y

t2 = Z * 2

t3 = t2 + A

t4 = t3 – t1

t5 = t4 + t3

Note: X, Y and Z are memory operands

R1 ← X

R2 ← Y

R1 ← R1 + R2

R2 ← Z

R2 ← R2 * 2

R3 ← A

R2 ← R2 + R3

R3 ← R2

R2 ← R2 − R1

R2 ← R2 + R3

Ans = 3

Question

Consider a register-memory architecture(2-address instructions supported) system. For this system the following intermediate code is going to be converted in machine code. Minimum how many registers are required in system so that the code can run without register spill?

t1 = X + Y

t2 = t1 – Z

t3 = t1 + t2

t4 = M + t3

t5= t2 – t4

R1 ← X

R1 ← R1 + Y

R2 ← R1

R1 ← R1 - Z

R3 ← R1

R1 ← R1 + R2

R2 ← M

R2 ← R2 + R1

R3 ← R3 – R2

Effective Address (E.A.)

Address of operand in a computation-type instruction or
The target address in a branch-type instruction.

Branch Instruction

Inst'n I2 is in execution in CPU

PC = 202

CPU decodes (detects) I2 as branch type instruction.

Condition

True

Branch taken

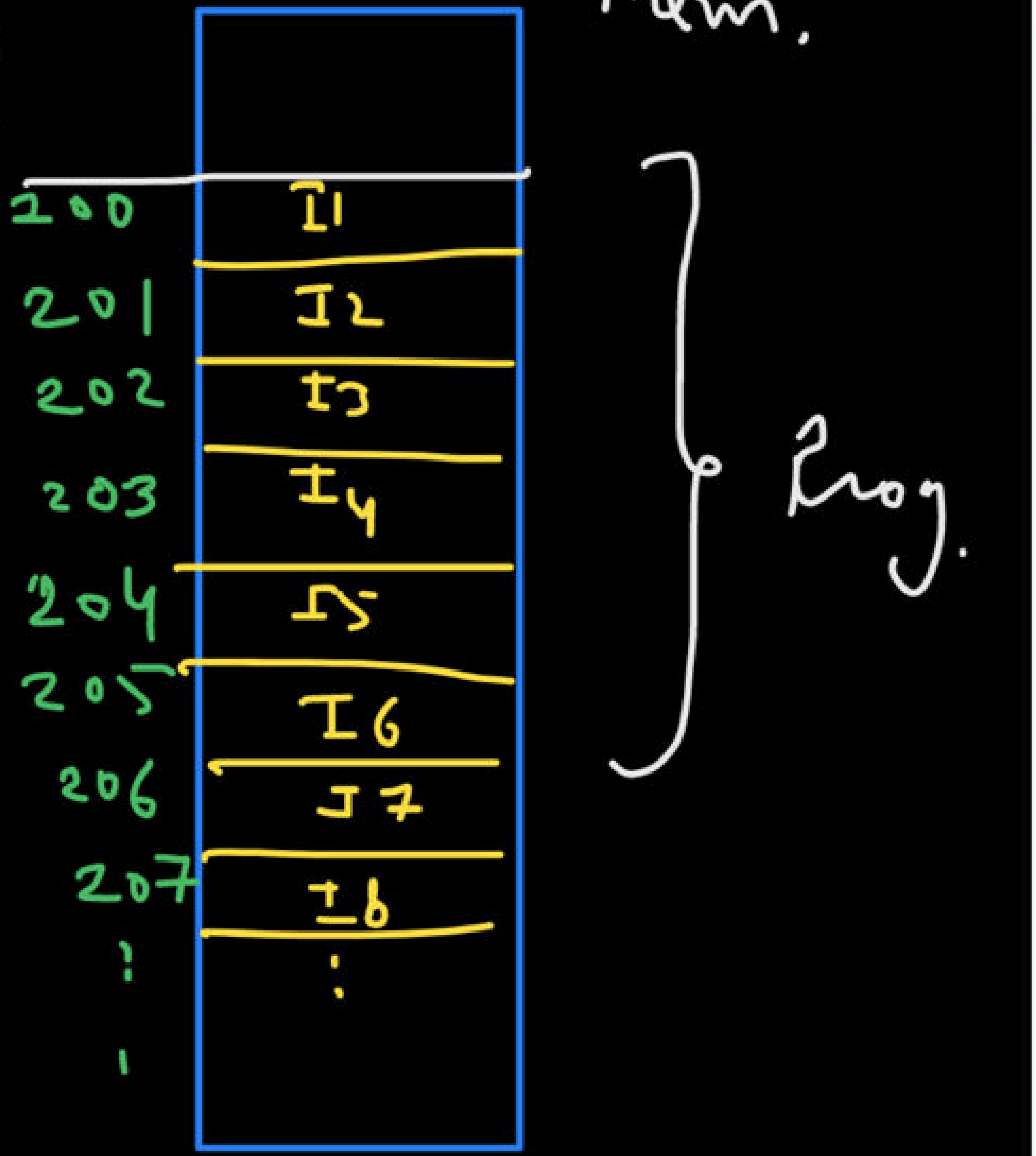
Target inst'n executed

PC = Target add (206) I7 target instn

False

Branch not taken

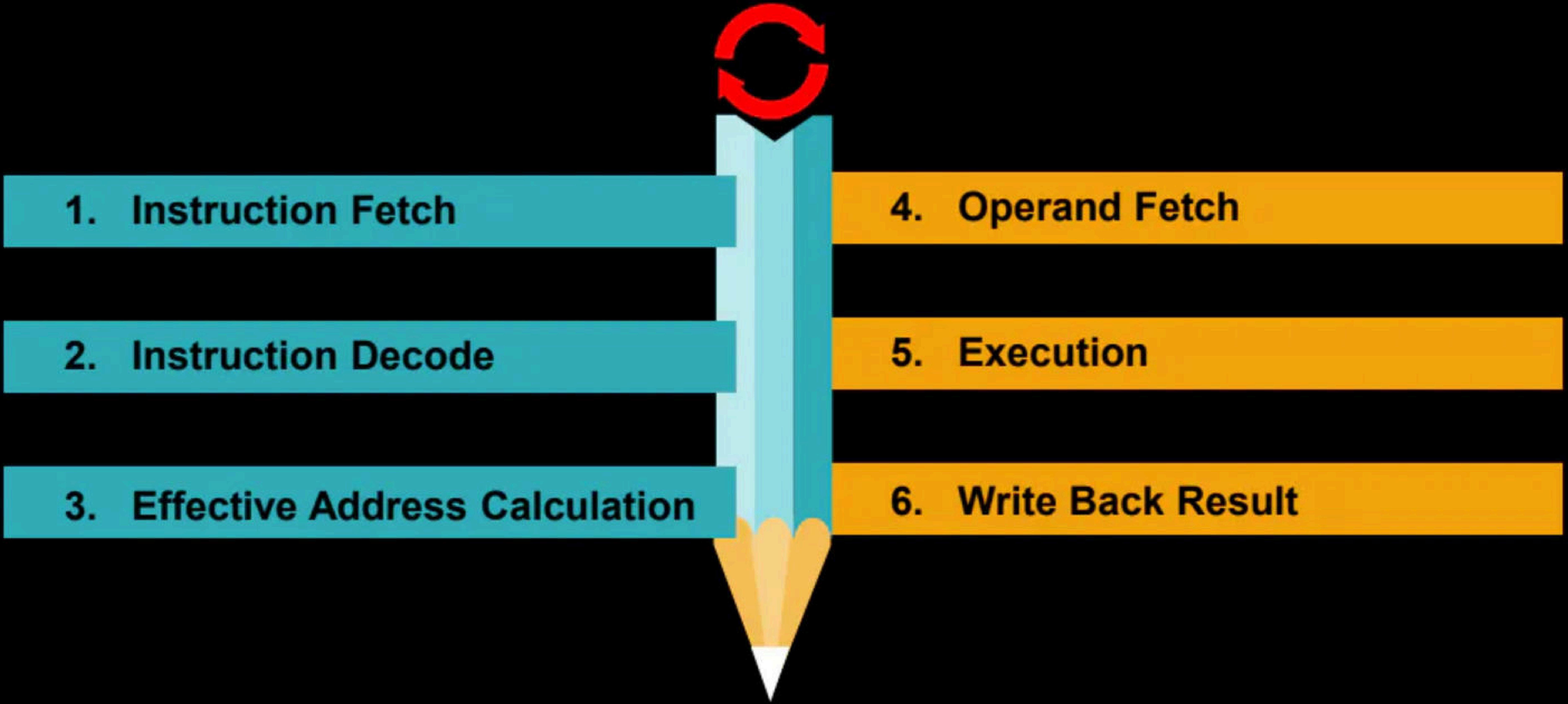
no change in PC value



200	I1
201	I2
202	I3
203	I4
204	I5
205	I6
206	I7
207	I8
:	:

Instruction Cycle

Instruction Cycle



1. **Instruction Fetch**
2. **Instruction Decode**
3. **Effective Address Calculation**
4. **Operand Fetch**
5. **Execution**
6. **Write Back Result**

Fetch Cycle & Execution Cycle

Computation vs Branch Type Instruction

Why Addressing Modes

Addressing Modes

It specifies how and from where the operands are obtained for an instruction

Happy Learning.!