TA calculation Problem

* Compute the TA for the following m/c instructions.

Given, X = 000690, B = 006030, PC = 003000

(I) 032600, (II) 03C300 (III) 022030 (IV) 010030 (V) 003600 (V) 0310C303

Hex — Binary — value loaded into reg. A

(I) 032600 → 000000 1 1 0 0 1 0 | 0110 0000 0000 | TA ⇒ 3600

(000000 = 0, 11 = 3, 0010 = 2 → F-3, 0110 0000 0000 = 6 0 0)

as it is PC relative, add PC value with disp. ∴ TA = disp + PC

(PC mode)

600 + 003000

= 3600

(II) 03C300 → 000000 1 1 1 1 0 0 | 0011 0000 0000 | TA ⇒ 6390

(F-3; 0011 0000 0000 = 3 0 0)

(indexed + base mode)

TA = (Base) + (X) + (disp)

006000 + 000090 + 300

= 6390

(III) 022030 → 000000 1 0 0 0 1 0 | 0000 0011 0000 | TA ⇒ 3030

(0000 0011 0000 = 0 3 0)

(indirect + PC mode)

TA = disp + PC

030 + 003000

= 3030

(iv) Hex

	op	n	i	x	b	p	e	disp/add	TA
010030 →	000000	0	1	0	0	0	0	0000 0011 0000	⇒ 30
								0 3 0	

(immediate addressing mode)

As i=1, it is immediate addressing mode, TA = displacement value.

TA = 30

(v) 003600

op	n	i	x	b	p	e	disp/add.	TA
000000	0	0	0	0	1	1	0110 0000 0000	⇒ 3600
							6 0 0	

(format-3 pc relative)

Here even though e=1 but - obj. code length is 6 digits, it is format-3. So displacement is there.

TA = PC + disp

= 003000 + 600 = 3600

(vi) 03 10C303

op	n	i	x	b	p	e	add	TA
000000	1	1	0	0	0	1	0000 1100 0011 0000 0011	⇒ C303
							0 C 3 0 3	

(format-4 object-code length is 8 digits & also e=1)

Here we dont have x, b, p as set to 1. ∴ TA = address.

TA = C303

Problems

Target-Address Calculation

Mode	Indication/Flags	TA calculation
(*) Base relative	b = 1, P = 0	TA = (B) + displacement (0 ≤ disp ≤ 4095)
(*) Program counter relative	b = 0, P = 1	TA = (PC) + displacement (−2048 ≤ disp ≤ 2047)
(*) Base/PC relative of Indexed also	x = 1, b or P = 1	→ TA = disp + PC or B + X

→ Flags/bits x, b, P : used to calculate the target-address using relative, direct- (b or P) and indexed addressing mode.

→ Flags/bits i and n : says how to use the target-address.

→ b and P : both if set to 0, disp field in format-3 instruction is taken to be the target-address.

(any of these addressing modes can also be combined with indexed addressing. if x = 1, x value is added for target-add. calculation)

For a format-4, b and P if 0, 20 bit-address is the target-address

(*) → x : if x is set-to 1, X register value is added for TA calculation.

→ $i=1, n=0$, immediate addressing.

(The operand value is already enclosed on the instruction (lies on last 12 or 20 bits)) TA: Target-address is used as the operand value (symbol address), no memory reference.

→ $i=0, n=1$, indirect-addressing

(The operand value points to an address that holds the address for the operand value) The word at the TA is fetched. value of TA is taken as the address of the operand value.

→ $i=0, n=0$ (or) $i=1, n=1$: Simple addressing (Direct-address)

TA is taken as the address of the operand value.

(x, b, P all set-to 0) : operand address goes as it is.