24.12.24

Final

Pin Description:

* $\overline{RD}$ ⇒ Active low
  - Read Activity
    - 0 (Read)
    - 1 (Inactive)

* $\overline{WR}$
  - 0 (Write)
  - 1 (Inactive)

Active High
- 0 (Inactive)
- 1 (active)

Active Low
- 0 (Active)
- 1 (Inactive)

* ALE (Address Latch Enable)
  - 1 (AD0 – AD15 ⇒ Address) [কখন data আর কখন address যাবে তা decide করবে]
  - 0 (AD0 – AD15 ≠ Address) [can be data or garbage]

* M/$\overline{IO}$ (↓ In/out memory)
  - 0 (I/O operation)
  - 1 (memory operation)

* **Reset :** কোনটা কাজ আবার প্রথম থেকে start করা এই কাজের জন্য 4টা clock cycle পর্যন্ত 1 দিতে হবে।

* **CLK :**

→ if high then normal signal. if low, then opposite signal →

* **VCC :** device ও Power supply এর জন্য use হয়।

* DT / $\overline{R}$ → 0 (Data Receive) => $\overline{R}$ pin
  - Data transfer
  - Receive
  - → 1 (Data transfer) => DT pin

* $\overline{DEN}$ → 0 (AD0 - AD15 => Data)
  - Data
  - Enable
  - → 1 (AD0 - AD15 ≠ Data) [address/spam but not data]

* $\overline{BHE}$ → 0 (D8 - D15 => Valid Data)
  - Bus High Enable
  - → 1 (D8 - D15 ≠ Valid Data)

D0 - D7 ↓ lower byte — Always Active

D8 - D15 ↓ upper byte

* A19/S6 - A16/S3: S → Status

***

8086 Bus Timing :

* **Machine Cycles:** A machine (bus) cycle consists of at least four clock cycles, called T states.

1. T1 :

① Send Address

② ALE pin active

③ M/$\overline{IO}$ → কোন কাজ করবে বুঝাতে

④ DT/$\overline{R}$ → transfer করবে নাকি receive করবে তা বুঝাতে

2. T2 :

① $\overline{RD}$ / $\overline{WR}$ → at a time, যেকোনো একটা use হবে

② $\overline{DEN}$ → enable করবে।

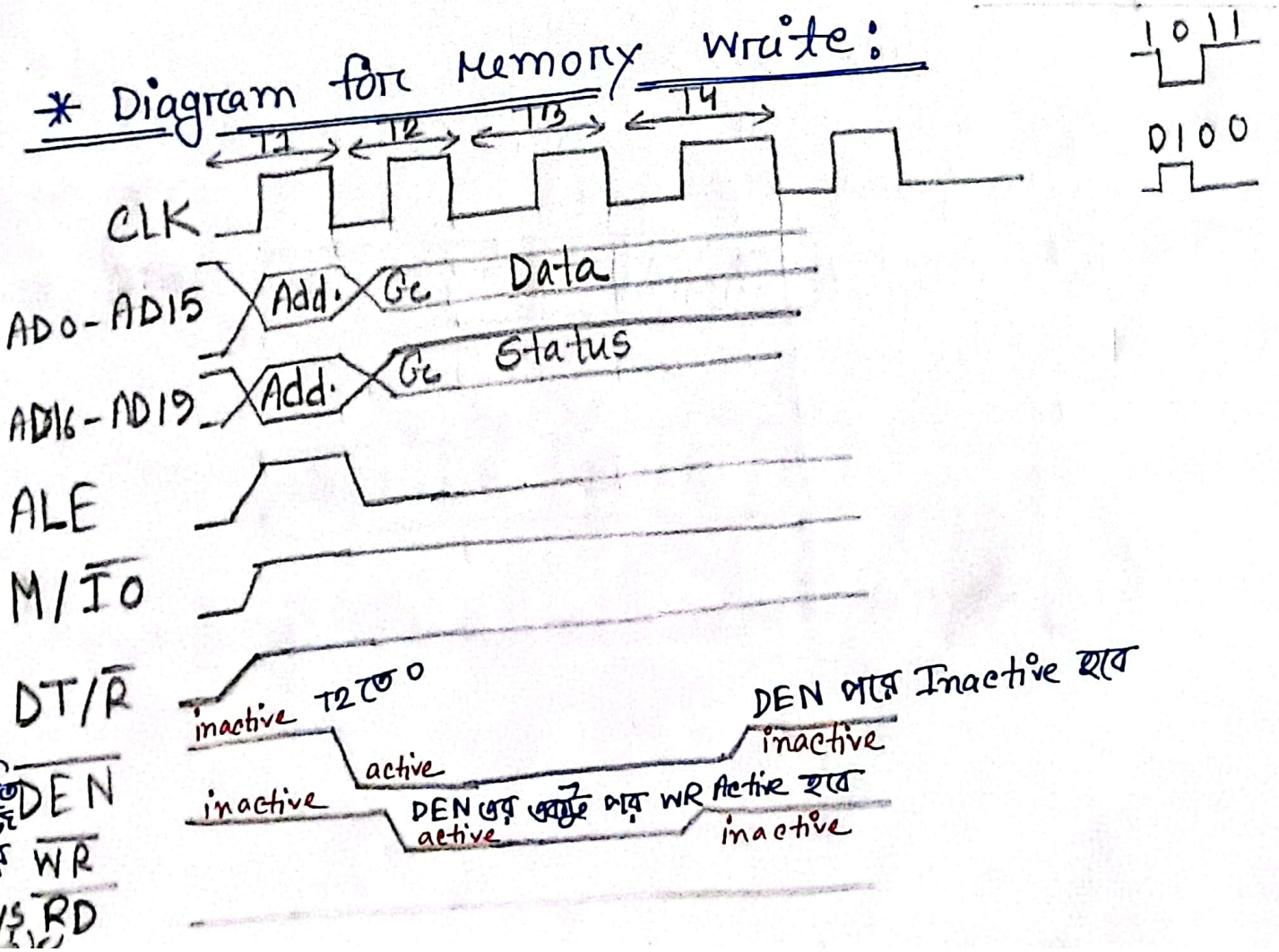
3. T3 :

① Send data

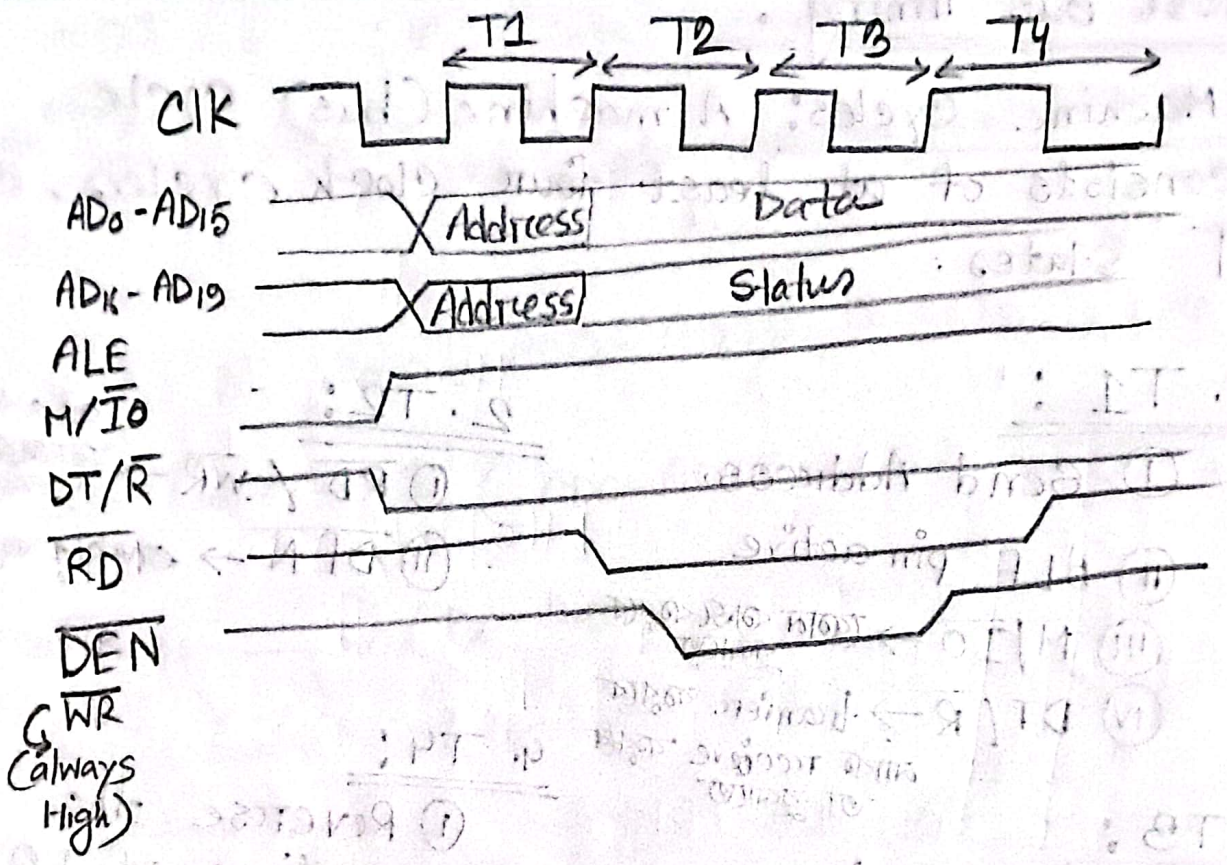
4. T4 :

① Reverse the operation of T2.

* **Diagram for Memory Write:**



* Diagram for Memory Read:



* Ready Pin: wait state → active. কাজ হবে না

28.12.24

Sequencial pipelining → 8086 MP fastest than any other MP.

*Operation number binary তে থাকবে।

Bus interface Unit (BIU) ← i) fetch: কোথায় location এর address or element নেবে বলতে হবে।

ii) Decode: let, add = 0010
sub = 0001

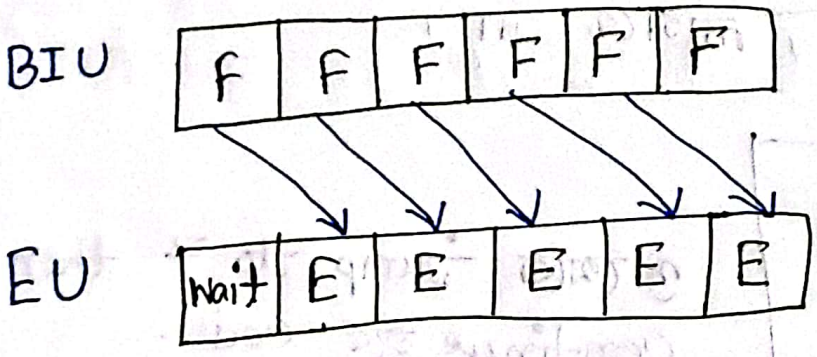
Execution Unit

ex:

OP	num	num
0010	0011	0001
Add	3	1

ex: 0001 0011 0
sub 3 1

iii) execute:



f = fetch

E = execution + Decode একসাথে

pipelining (fetch and execution cycle) in 8086.

→ 4/5 marks

* 2 cases for this :

mov → Value store করবে Ax-এ

mov AX, 5H → Value

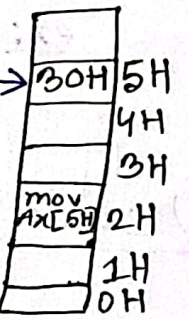
⇓

Ax = 5H

mov AX, [5H] → Address / location

⇓

Ax = 30H

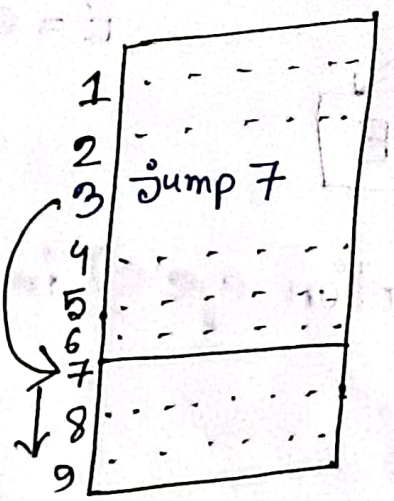


(Address এর মধ্যে যেই value থাকবে তা বসবে)

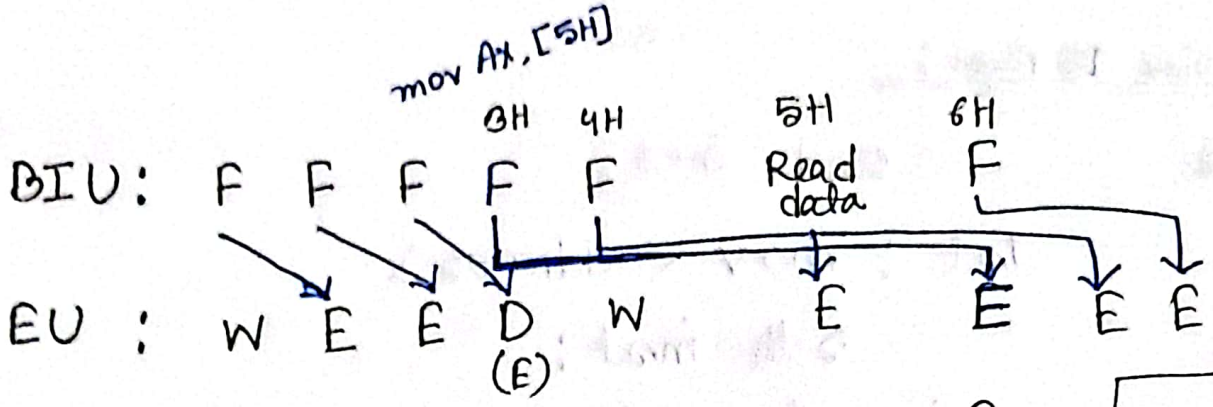
jump : 2n inst :

jump <5th>

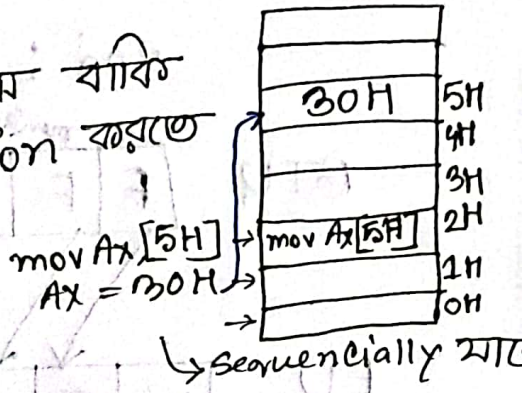
* jump করে আসা location-এ '*' বসবে।
jump-এ jump করে গেলে next-এ যেতে থাকবে।
থাকের ঝুলো লাগবে না।



3 থেকে jump to 7 then 8 থেকে continue হবে code.

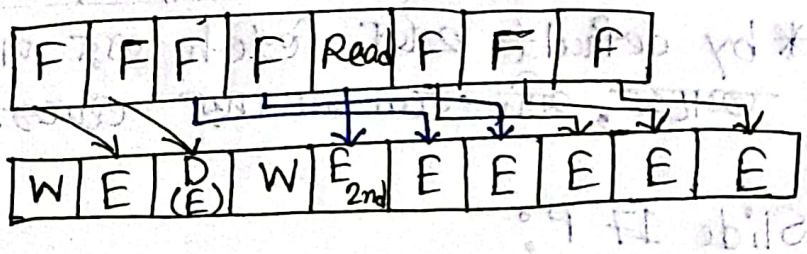


* mov এর ক্ষেত্রে যেই fetch আসে বাকি ছিল condition শেষে তার execution করতে হবে।



* 1st case : mov. instruction

Slide 15P b) (1st half)
ans :

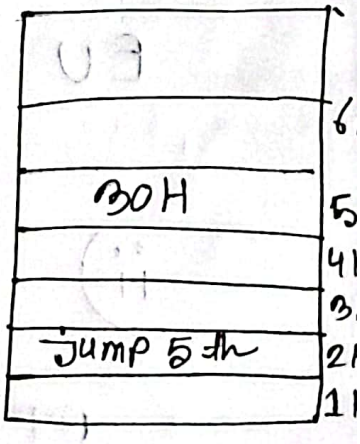


2nd case : jump instruction

2nd inst : jump <5th>

BIU :

EU :



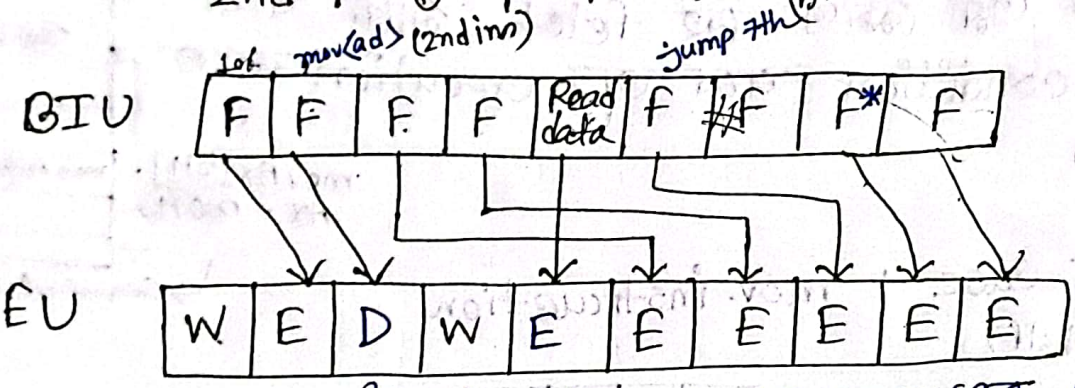
Slide 15 Page:

b) 2nd inst:

1st : mov <address>

5th inst:

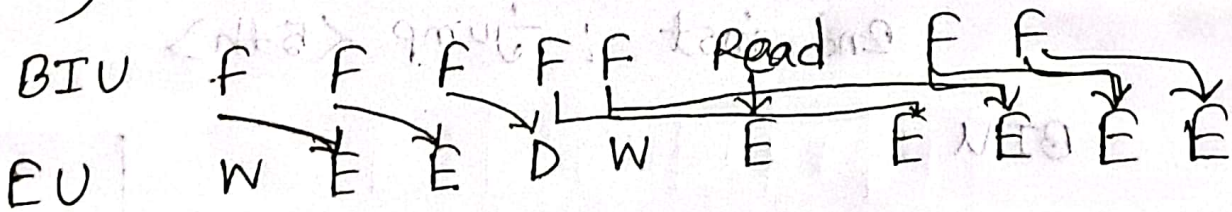
2nd : jump <7th>



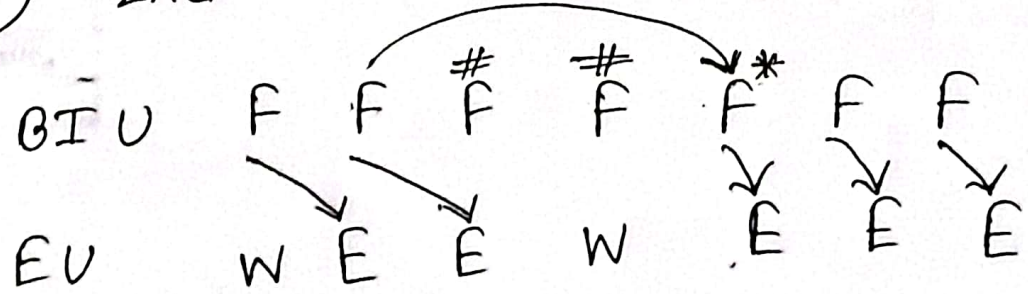
*by default 2টার fetch এর পর mov এর Read data আসে, যদি বলে না দেয়া Question এ।

Slide 17 P:

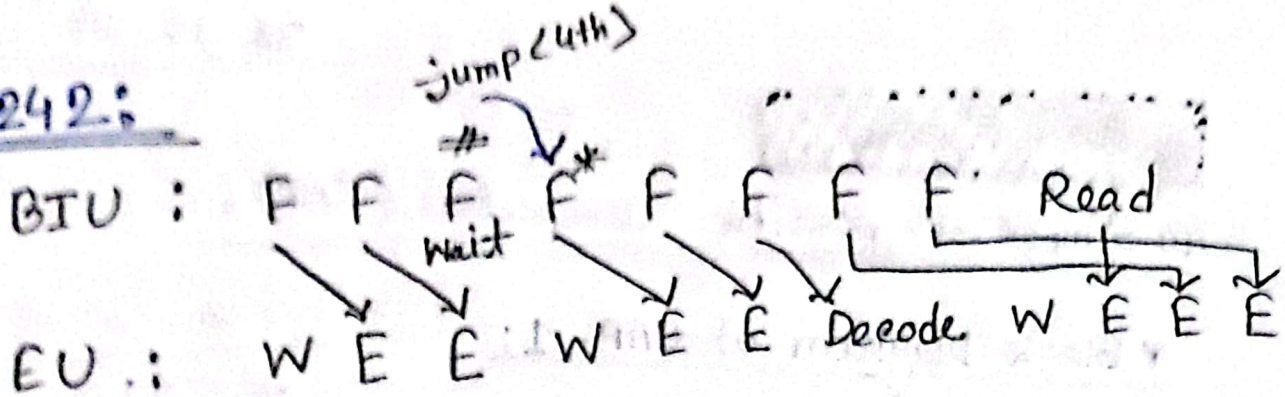
c) i) 3rd inst mov <add>



ii) 2nd inst jump 5th

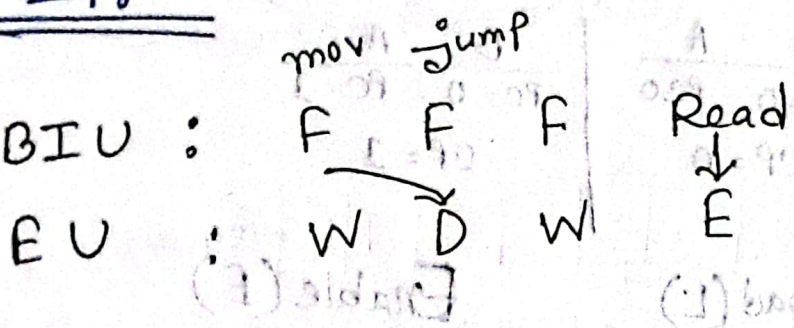


*242:



* Spring-24:

3.a)



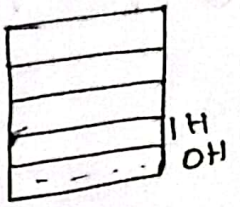
* mov and memory read operation same नाही
* Pipeline break - jump, mov, or (mov) memory address

31.12.24

Simple as possible Computer (SAP): → 10 Marks

* Block Diagram of SAP-1:

① Program Counter: PC = 1H



→ Inside the PC, $C_P = 0$ (No change in PC)

$C_P = 1$ (PC++)

B	A
PC=0	PC=0
$C_P = 0$	

B	A
PC=0	PC=1
$C_P = 1$	

⊛ load (L) (Bus → Device) → Active High

Enable (E) (Device → Bus)

⇒ $E_P = 0$ (Inactive)

$E_P = 1$ (PC → Bus)

* Clk → active low (signal pass করবে যার উপর base করে কাজ করতে থাকবে)

* CLR → active (value detect করে 0 করে দিবে)

② Input & MAR: (MAR → Memory Address Register)

→ Active low

⇒ $\overline{L_M} = 0$ (Bus → MAR)

low, MAR

$\overline{L_M} = 1$ (Inactive)

PC তার কাছে যেই address থাকবে তা MAR তার কাছে চলে যাবে। MAR তা RAM-এ pass করে।

③ Ram: $\overline{CE} = 0$ (RAM → Bus)

$\overline{CE} = 1$ (Inactive)

Chip enable

$16 \times 8 \rightarrow D.B = 8$

$2^n = 16$
$= 2^4$ $\therefore n = 4 = A.B$

(IV) Instruction Register: RAM থেকে 8 bit value নিয়ে 4 bit -এ devide করে নেয়।

$\Rightarrow \overline{L_i} = 0$ (Bus → Instruction Reg.)

$\overline{L_i} = 1$ (Inactive)

$\Rightarrow \overline{E_i} = 0$ (Instruction Reg → Bus)

Enable ↓ ins.reg.

$\overline{E_i} = 1$ (Inactive)

* Value রাখার জন্য Register:

(I) Accumulator: Short form AX

$\Rightarrow \overline{L_A} = 0$ (Bus → AX)

$\overline{L_A} = 1$ (Inactive)

$\Rightarrow E_A = 0$ (Inactive)

$E_A = 1$ (AX → Bus)

(II) Adder/Subtracter: S_u = Subtraction Unit

$\Rightarrow S_u = 1$ (Subtract)

$S_u = 0$ (Add)

$\Rightarrow E_u = 1$ (Adder/Substractor → Bus)

$E_u = 0$ (Inactive)

(iii) B Register; register to store value.

$\Rightarrow \overline{L_B} = 0$ (Bus → B Register)

$\overline{L_B} = 1$ (Inactive)

(iv) Output Register;

$\Rightarrow \overline{L_O} = 0$ (Bus → Output Reg.)

active low → 0 হলে active

$\overline{L_O} = 1$ (Inactive)

Computer can complete ৯ Instructions:

1. LDA (Load Accumulator):

LDA <memory address> (সিন্ট্যাক্স নিয়ম)

আগের value remove হয়ে new value store হয়।

$\Rightarrow$ LDA 0H

$\Downarrow$

AX = 20H

$\Rightarrow$ LDA 3H $\Rightarrow$ AX = 7H

[LDA দিলে আগের value remove হয়ে new value store হয়]

4H	4H
7H	3H
5H	2H
10H	1H
20H	0H

RAM

2. ADD : ADD <memory address>

$\Rightarrow$ ADD 2H

$\Rightarrow$ AX = AX + value of (2H)

= 7H + 5H

= CH

3. SUB : SUB 4H

(4H → address)

$\Rightarrow$ AX = AX − value of (4H)

$\Rightarrow$ AX = CH − 4H

= 8H

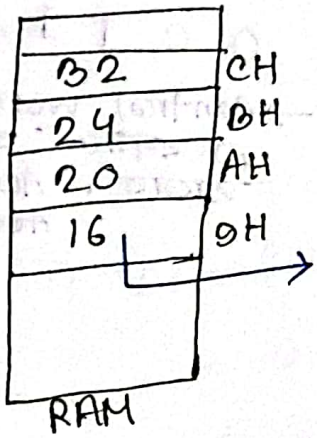
4. OUT: Out Register = accumulator

Out Reg. = 8H [it will pass the output directly]

5. HLT: Stops clock (Halt assembly)

(Shut down or stop all the clk in the computer)

Q: We Want to complete 16+20+24-32.



→ Hexa decimal - ও প্রথমে decimal - ও convert করতে হবে (ex: 10H → 16)

ans:

	AX	BX
LDA 9H	AX=16	BX=0
ADD AH	AX=16+20 =36	BX=20
ADD BH	AX=36+24 =60	BX=24
SUB CH	AX=60-32 =28	BX=32
Out	AX=28	BX=32
HLT	AX=0	BX=0

*Convert To Machine Code =>

```
0000 1001 ← LDA 9H
0001 1010 ← ADD AH
0001 1011 ← ADD BH
0010 1100 ← SUB CH
1110 XXXX ← Out
1111 XXXX ← HLT
```

CamScanner

04.01.2425

Q: machine code → Assembly code
Assembly → Binary
Binary → Assembly
ex: 0000 0111 → LDA 7H

Control Word:

$CON = C_P E_P \overline{L_M} \overline{C_E}\ \overline{L_I} \overline{E_I} \overline{L_A} E_A\ S_U E_U \overline{L_B} \overline{L_O}$

$= 0\ 0\ 1\ 1\ \ 1\ 1\ 1\ 0\ \ 0\ 0\ 1\ 1$

$= 3E3\ H$

= INACTIVE (NOP)

→ Control word কোনো কাজ না করালে সব pin inactive থাকবে। Active high → 0 (in.) Active low → 1 (in.)

*For LDA:

Fetch Cycle (LDA, ADD, SUB, OUT) 3 cycle - 9 time

T1:
PC → Bus
Bus → MAR

$CON = C_P E_P \overline{L_M} \overline{C_E}\ \overline{L_I} \overline{E_I} \overline{L_A} E_A\ S_U E_U \overline{L_B} \overline{L_O}$

$= 0\ 1\ 0\ 1\ \ 1\ 1\ 1\ 0\ \ 0\ 0\ 1\ 1$

$= 5E3\ H$

T2:
PC++

$CON = C_P E_P \overline{L_M} \overline{C_E}\ \overline{L_I} \overline{E_I} \overline{L_A} E_A\ S_U E_U \overline{L_B} \overline{L_O}$

$= 1\ 0\ 1\ 1\ \ 1\ 1\ 1\ 0\ \ 0\ 0\ 1\ 1$

$= BE3\ H$

T3:
RAM → Bus
Bus → Ins. Reg.

$CON = C_P E_P \overline{L_M} \overline{C_E}\ \overline{L_I} \overline{E_I} \overline{L_A} E_A\ S_U E_U \overline{L_B} \overline{L_O}$

$= 0\ 0\ 1\ 0\ \ 0\ 1\ 1\ 0\ \ 0\ 0\ 1\ 1$

$= 263\ H$

T4:
Ins. Reg → Bus
Bus → MAR

$CON = C_P E_P \bar{L}_M \bar{C}_E \; \bar{L}_I \bar{E}_I \bar{L}_A E_A \; S_U E_U \bar{L}_B \bar{L}_O$
$= 0\;0\;0\;1 \quad 1\;0\;1\;0 \quad 0\;0\;1\;1$
$= 1A3\ H$

T5:
RAM → Bus
Bus → AX

$CON = C_P E_P \bar{L}_M \bar{C}_E \; \bar{L}_I \bar{E}_I \bar{L}_A E_A \; S_U E_U \bar{L}_B \bar{L}_O$
$= 0\;0\;1\;0 \quad 1\;1\;0\;0 \quad 0\;0\;1\;1$
$= 2C3\ H$

T6:
INACTIVE

For ADD: T1, T2, T3, T4 → Same as LDA

T5:
RAM → Bus
Bus → BX

$CON = C_P E_P \bar{L}_M \bar{C}_E \; \bar{L}_I \bar{E}_I \bar{L}_A E_A \; S_U E_U \bar{L}_B \bar{L}_O$
$= 0\;0\;1\;0 \quad 1\;1\;1\;0 \quad 0\;0\;0\;1$
$= 2E1\ H$

T6:
~~RAM~~ ADD
Adder / sub → Bus
Bus → AX

$CON = C_P E_P \bar{L}_M \bar{C}_E \; \bar{L}_I \bar{E}_I \bar{L}_A E_A \; S_U E_U \bar{L}_B \bar{L}_O$
$= 0\;0\;1\;1 \quad 1\;1\;0\;0 \quad 0\;1\;1\;1$
$= 3C7\ H$

For Sub:

T6:
Adder / Sub → Bus
Bus → Ax

$CON =$
$= 0\;0\;1\;1 \quad 1\;1\;0\;0 \quad 1\;1\;1\;1$
$= 3Cf\ H$

FOR OUT:

CON=

= 0011 1111 0010

= 3F2 H

T4:
AX → Bus
Bus → OUT Reg.

T5:
INACTIVE

T6:
INACTIVE

* Summer-24:

$3^2 - 2^2 + 1^2$

$= 3 \times 3 - (2 \times 2) + 1$

$= 3 + 3 + 3 - 2 - 2 + 1$

Address

0H ← LDA F9H (3H) 1001 1111

1H ← ADD FH

2H ← ADD FH

3H ← SUB CH

SUB CH

ADD DH

OUT ⟶ 0100 XXXX

HLT ⟶

*Spring - 24:

0H → 1010 1010 → LDA AH → AX / 32

1H → 1100 1100 → SUB CH → 32 - 10 = 22

4H → 1111 XXXX → OUT

যদি BX, OUT বের করতে বলে,

BX	OUT
0	0
10	0

c) RAM

256 X 16 → D.B = 16 bit

$2^n = 256$

$= 2^8$

$\therefore n = 8$

$\therefore A.B = 8$ bit

07.01.25

Assignment :

4. LDB_{OUT}

1. RAM → B Reg.
2. B Reg → OUT

T4: 1. Ins. Reg. → Bus
2. Bus → MAR

$CON = C_P E_P \bar{L}_M \bar{C}E \quad \bar{L}_I \bar{E}_I \bar{L}_A E_A \quad S_U E_U \bar{L}_B \bar{L}_O$

$= 0\ 0\ 0\ 1 \quad 1\ 0\ 1\ 0 \quad 0\ 0\ 1\ 1$

$= 1A3H$

T5: 1. RAM → Bus
2. Bus → B Reg.

$CON = C_P E_P \bar{L}_M \bar{C}E \quad \bar{L}_I \bar{E}_I \bar{L}_A E_A \quad S_U E_U \bar{L}_B \bar{L}_O$

$= 0\ 0\ 1\ 0 \quad 1\ 1\ 1\ 0 \quad 0\ 0\ 0\ 1$

$= 2E1H$

T6: 1. Adder → Bus
2. Bus → Out Reg.

$CON = C_P E_P \bar{L}_M \bar{C}E \quad \bar{L}_I \bar{E}_I \bar{L}_A E_A \quad S_U E_U \bar{L}_B \bar{L}_O$

$= 0\ 0\ 1\ 1 \quad 1\ 1\ 1\ 0 \quad 0\ 1\ 1\ 0$

$= BE6H$.

CamScanner

Communication Protocols : (Part-1)

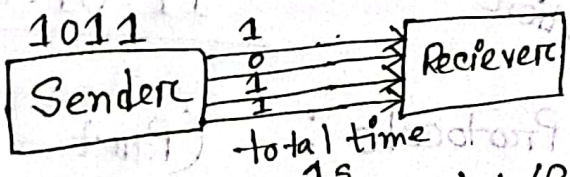
2 types :

① Serial → একটার পর একটা কাজ হবে।
Serially data transfer হবে।

Sender → Reciever
1s
1011 (total 4s)

② Parallel → একসাথে অনেকগুলো কাজ হবে। (faster)
multiple wire থাকবে।

costly

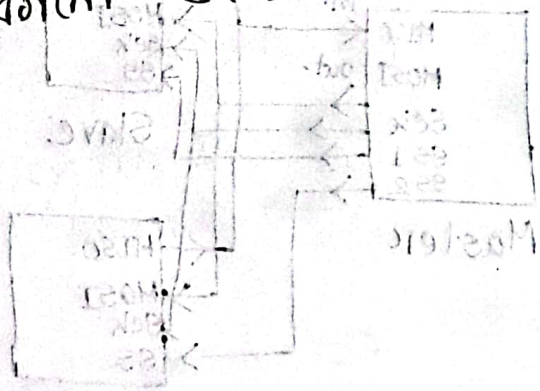


*Situation অনুযায়ী Serial/Parallel use করব।

2 Problems in Parallel Communication :

1. Clock Skew : কোনো একটা fault এর কারণে time বেশি লাগবে।

2. Crosstalk :

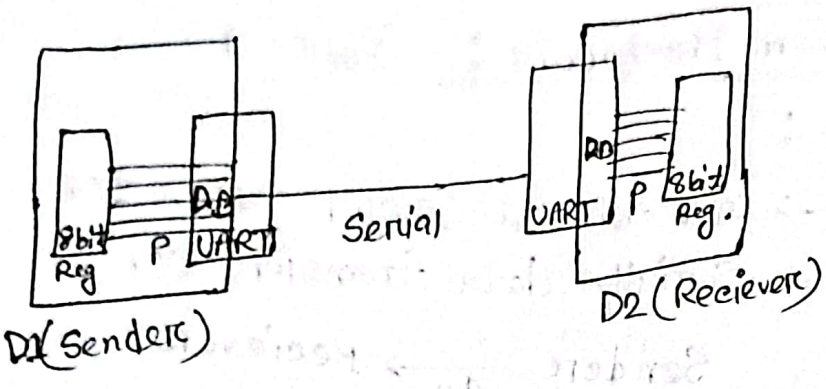


UART : It is a device.

↳ Universal Asynchonous Receiver Transmitter

UART device এর main কাজ হচ্ছে parallel data-কে Serial data-তে বা Serial data কে Parallel data-তে Convert করা।

CamScanner



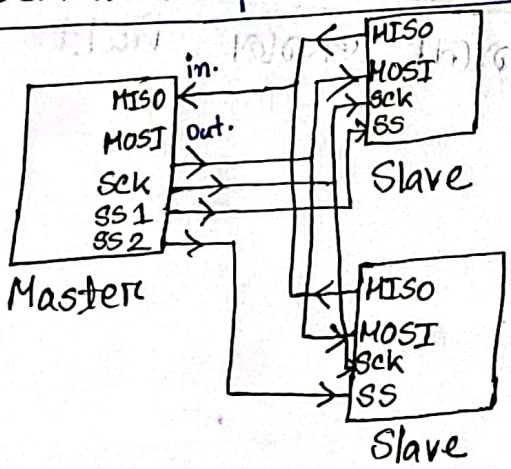
→ diagram related question

11.01.25

Communication Protocols : (Part - 2)

Two types of protocols :

① Serial Peripheral Interface (SPI): (3 wire based)



(1 master device remaining all slave device)

MISO → Master in slave out
MOSI → Master out slave in
SCK → Serial clk
SS/CS → Slave select / Chip select

* যতগুলো Slave থাকবে master-এ ততগুলো PIN থাকবে
* PIN বেশি থাকলে cost বেড়ে যাবে (problem)
* master device-এ কোনো problem হলে পুরো system নষ্ট হয়ে যায় (problem)
* যখন যেই Slave এর আগে connect হবে সেটা 1 হবে, বাকি SS/CS 0 হবে।

(11) I2C protocol :

1024 টি থাকলে $\Rightarrow 2^{10}$

একটি PIN দুটি slave control করতে পারবে। So, 2^n use করব।

n = master device PIN

2^n = slave count

SCL → Serial clock

SDA → Serial Data (Bi-directional) (master পাঠাবে)

* How it works :

① Start bit / start condition পাঠাতে হবে।

Handshaking Protocol

② address দিবে (7 or 10 bit)

③ decide read / write (master দিবে)

* WRITE → 0

* READ → 1

④ Send Ack/NAck to master (Slave পাঠাবে master কে)

⑤ 8 bit করে data transfer হবে। every time 8 bit করে fixed.

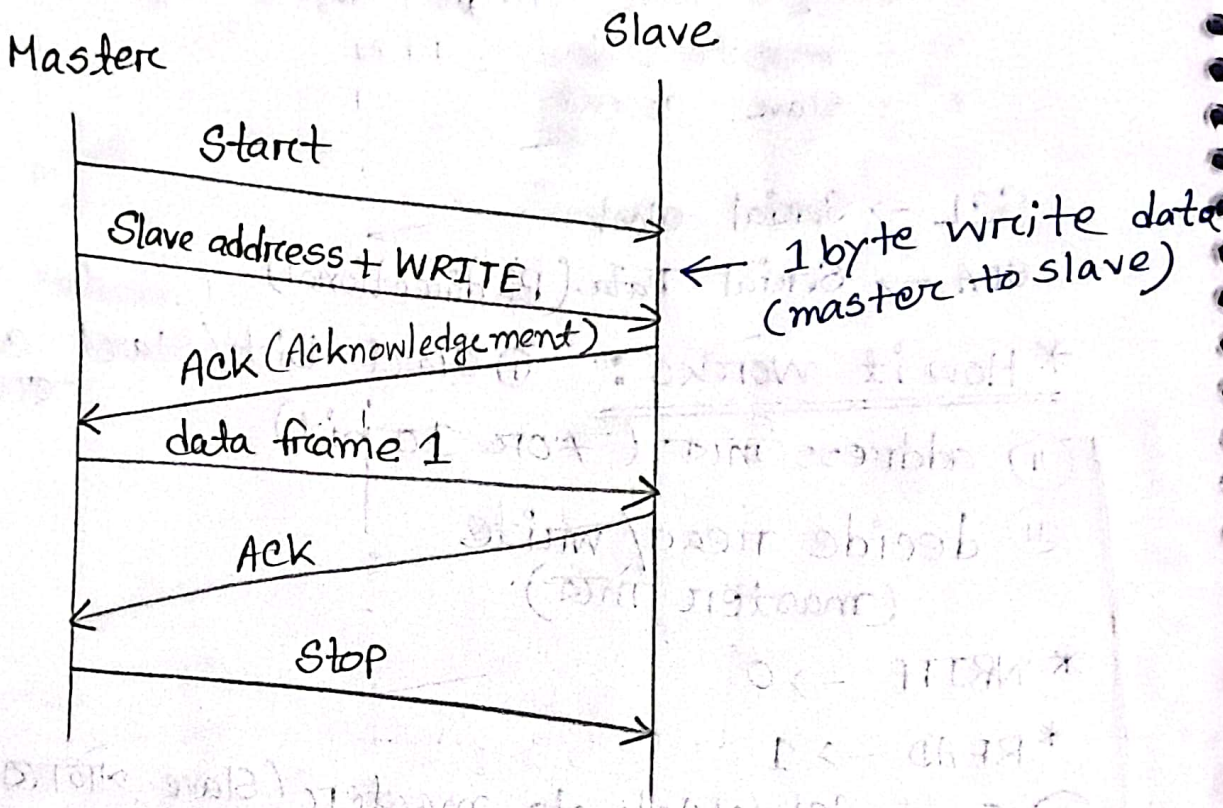
• 32 bit থাকলে 4 বার পাঠাবে।

⑥ 8 bit পর পর every time Ack / NAck পাঠাবে। (Slave to master)

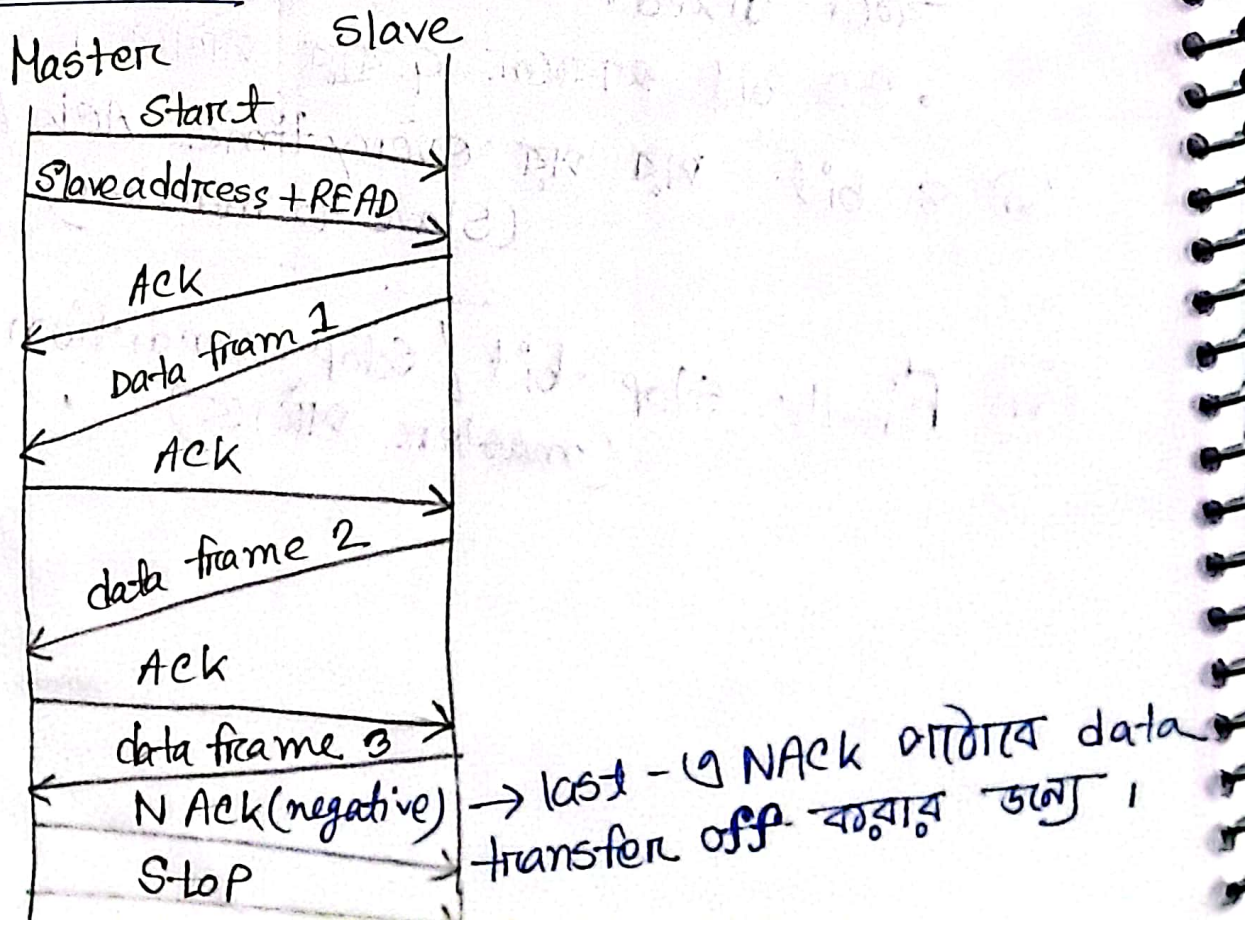
⑦ Finally stop bit / stop condition পাঠাবে। (master পাঠাবে)

CamScanner

* Diagram: (Write)

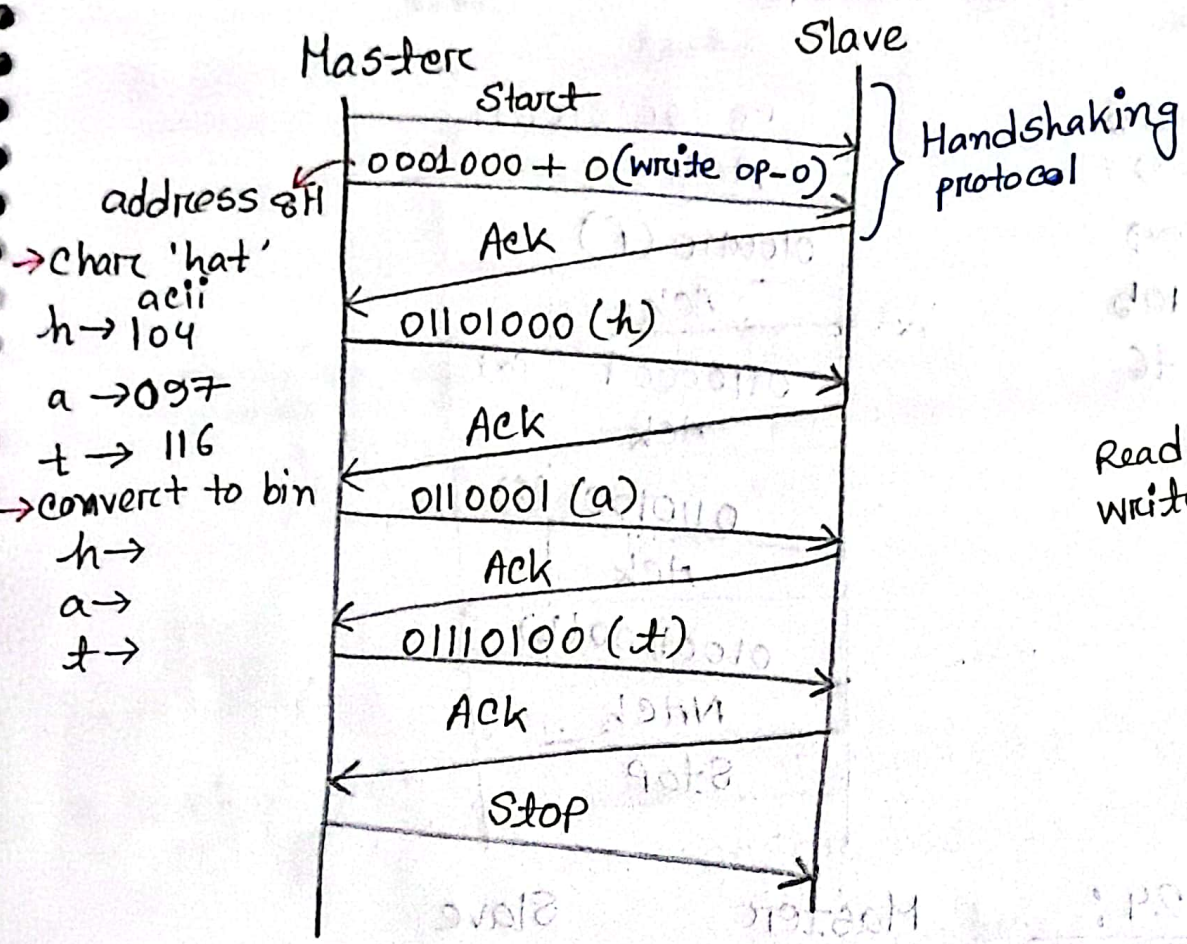


* Diagram (Read):



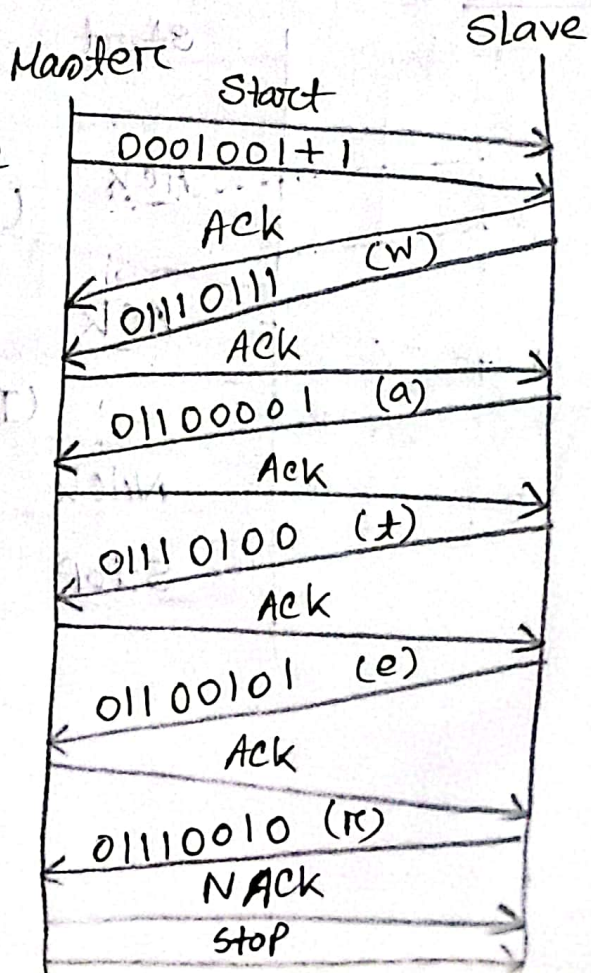
*Slave এর address
= 7 bit বা 10 bit

*Exercise: Page-21



* Page-22:

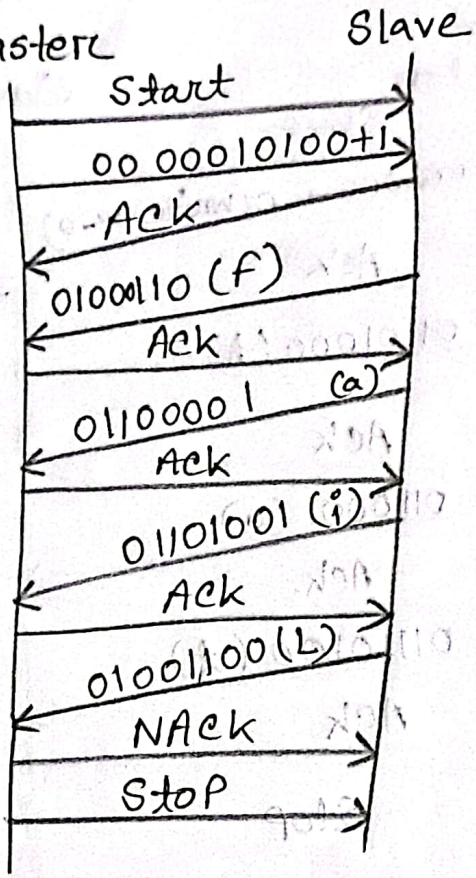
* Master recieve 5byte data char ("Water") from slave 9H.



14.01.25

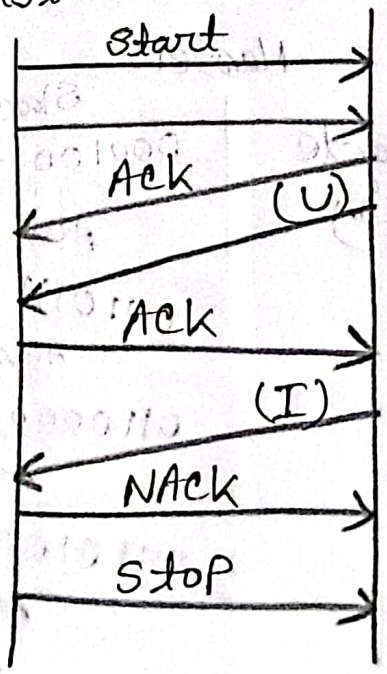
*** 242 :** 2. b) Master Slave

A = 65
a = 97
c = 99
i = 105
L = 76



*** Spring - 24 :**

Master Slave



*233: 2.c) D2H কে direct binary তে নিব।

Difference Between MP and Micro-controller:

μP + RAM + I/O
↓
μc

* Arduino and Raspberry pi:

→ Arduino (Micro-controller based)
→ Raspberry pi (μp based)

*242: c) Arduino

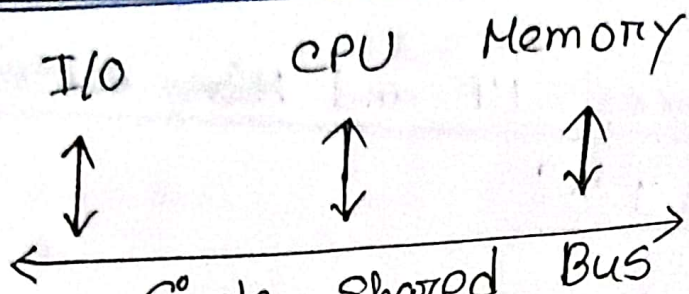
*Spring-24: b) home security → Raspberry pi
sensor node → Arduino

*233: b) 1st task → Arduino
2nd task → Raspberry pi

Harvard Architecture:

program memory ⟷ CPU ⟷ Data memory

Von Neuman Architecture:

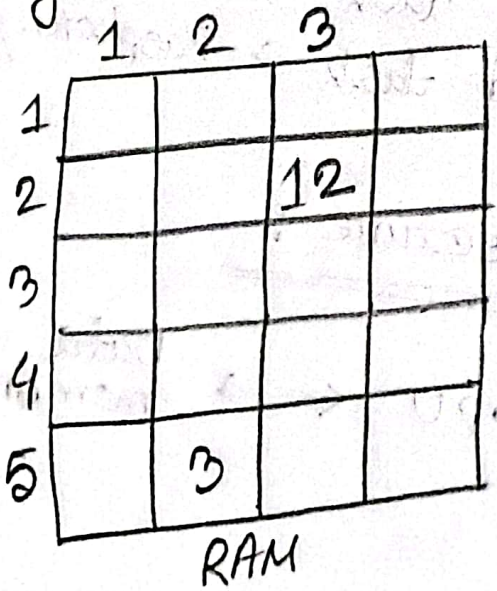


→ Cost কমবে But, Slower.

RISC vs CISC:

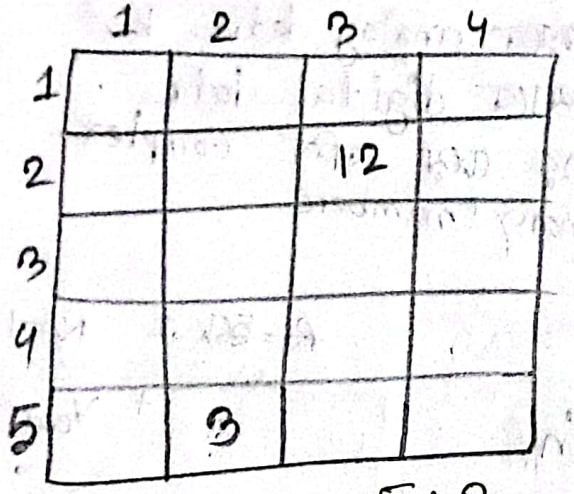
* RISC → Reduced Instruction set Computer
* CISC → Complex Instruction set Computer

Q: given MULT 2:3, 5:2 for CISC



For RISC,

Load A, 2:3
Load B, 5:2
PROD A, B → A = AXB => 4X3 = 12
STORE 2:3, A



op code = 4 bit
Reg = " "
memory = 8 bit

MULT 2:3 5:2 => 4+8+8 = 20 bit
↓ ↓ ↓
op. mem. mem.

MULT A , 2:3 => 4+4+8 = 16 bit
↓ ↓ ↓
op.. Reg. mem.

MULT A, B => 4+4+4 = 12 bit
↓ ↓ ↓
op. Reg. Reg.

* বড় instruction নিয়ে ছোট্ট Register use → CISC
* ছোট্ট " " বড় " → RISC

Analog to Digital & Digital to analog Conversion:

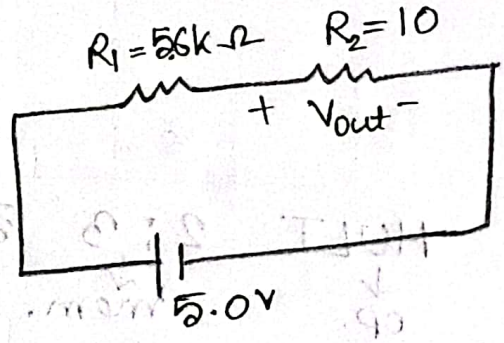
Analog → Continuous

* Types of Sensor:

(i) Analog → room এর temp (fixed না)

(ii) Digital → read করবে analog data but out দেখাবে digital data.
Analog এর চেয়ে বেশি complex.
ex: binary number

* $V_{out} = \frac{R_2}{R_1 + R_2} \times V_{in}$



$V_{out} = 0.76\ V$

* Page - 38:

$R_1 = 5.6\ k\Omega$

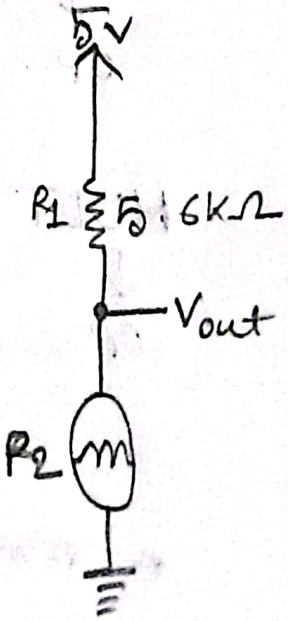
$V = 5\ V$

$V_{out} = 2.78\ V$

$R_2 = ?$

now, $V_{out} = \frac{R_2}{R_1 + R_2} \times V_{in}$

$\Rightarrow$ =



Ans: 7.01kΩ

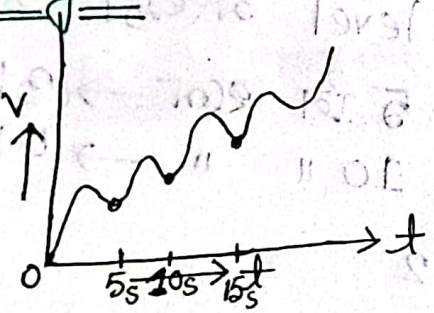
20.01.25

Analog to Digital Converter (ADC):

ADC এর জন্য যে technique use করা হয় সেটা হল Pulse-Code modulation (PCM).

- 3 steps for ADC =>
  - (I) Sampling
  - (II) Quantization
  - (III) Encoding

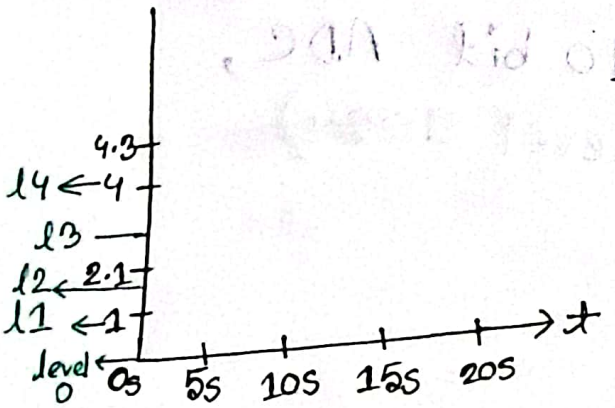
(I) Sampling:



→ Sampling এর ক্ষেত্রে, নির্দিষ্ট time পর পর value নিব।
→ accuracy-এর জন্য time interval কমিয়ে নিব।
→ time interval বাড়িয়ে দিলে কম data নিয়ে কাজ করা যাবে।

(II) Quantization:

→ Quantization এর ক্ষেত্রে, fractional value, remove করতে হয়।
→ Closest number এর সাথে map করে নিব।



(0,0), (5, 4.3), (10, 2.1), (15, 1), (20, 4)

→ (5, 4.3) → level 4
→ (10, 2.1) → level 2
} remove all the fractional number

→ 0 → level 0
0.5 → level 1
⋮
2 → level 4
} level বাড়িয়ে gap কমালে error কমবে। accuracy বাড়বে। But অনেক Space লাগবে। processing speed কমে যাবে। So, level কম রাখব।

*Quatization Error: The difference between

(iii) Encoding:

0 → level 0 → 000
0.5 → level 1 → 001
1 → level 2 → 010
1.5 → level 3 → 011
2 → level 4 → 100

এখানে level ৫টা তাই binary তে 3 bit নিয়েছি

→ level গুলোকে binary-তে convert করাই encoding এর কাজ।

encoding করার জন্যে level সংখ্যা জানতে হবে। যেমন: level ৫টা হলে → 3 bit
level 10 " " → 4 bit

or, 1 bit → 2
2 bit → 4
3 bit → 8 (2^3)
⋮
10 bit → 1024 (2^{10})

Q: We are using 10 bit ADC,
0 - 15 (total level 1024)

0 → level 0
15 → level 1023

*Page-28: The 10 bit ADC of ..... 5v System. if analog voltage is 2.12v then what will be the ADC report as a value?

ans: 10 bit ADC → 1024 (5v)
↳ maximum 5v measure করবে।

5v → level 1023
1v → level $\frac{1023}{5}$
2.12v → level $\frac{1023 \times 2.12}{5}$
$= 433.752 \approx$ level 434/43

formula: $\frac{\text{resolution of the ADC}}{\text{System voltage}} = \frac{\text{ADC reading}}{\text{Analog voltage measured}}$
↳ মেপে বের করবে

Digital to Analog Converter (DAC):

DAC তে আগের মত exact value পাওয়া যায় না।

→ DAC এর জন্য যে technique use করি সেটা হচ্ছে pulse width Modulation (PWM).

→ কতক্ষণ 0 দিব আর কতক্ষণ 1 দিব সেটা হচ্ছে "Duty cycle".

* Page - 36:

12 bit → 2^{12} ⇒ 4096 generate হবে 0-4095

analogWrite(0) → duty cycle 0%

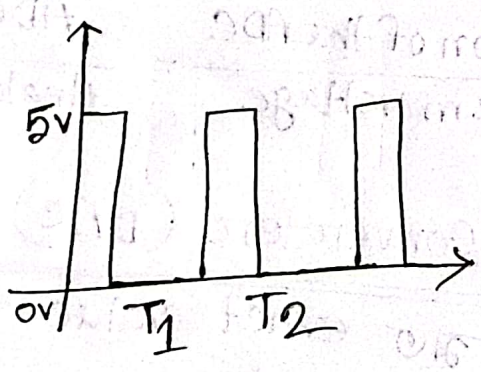
" (4095) → " 100%

" (1) → " 100%/4095

" (1024) → " $\frac{100\% * 1024}{4095}$ = 25%

Arduino → ADC → 10 bit
Arduino → DAC → 8 bit

Fixed



1 সিব 25% আর বাকি time 0 সিব।

21.01.25

*242: 2.a) $\frac{2^n - 1}{5} = \frac{2457}{3}$

$\Rightarrow 2^n - 1 = 4095$

$\Rightarrow 2^n = 4096$

$\Rightarrow n = \log(4096)$

$\therefore n = 12$

here, level বাড়াতে হবে (Quantization error কমানোর জন্য)

*Spring-24: 2.a) 1023 — 5V

1 — $\frac{5}{1023}$ V

$\therefore$ 934 — $\frac{5 \times 434}{1023}$ V

$= 2.12$ V

b) 100% DC → analogWrite(4095)

$\therefore$ 1% " → $\frac{4095}{100}$

$\therefore$ 75% " → $\frac{4095 \times 75\%}{100\%}$

$= 3071.25$

$\approx 3071\%$

*233:

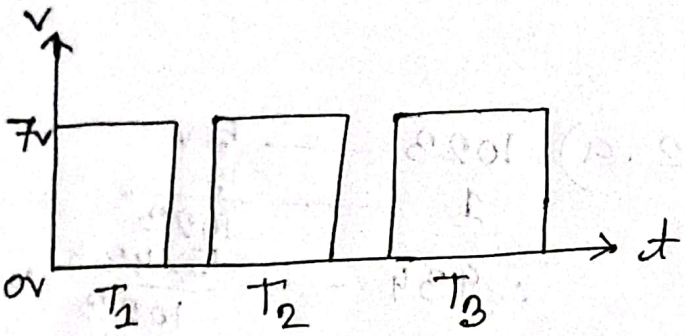
2.a) 11 bit ($2^{11} = 2048$)
(0 - 2047)

~~analogWrite (1) → 100%.~~

analogWrite (2047) → duty cycle 100%.

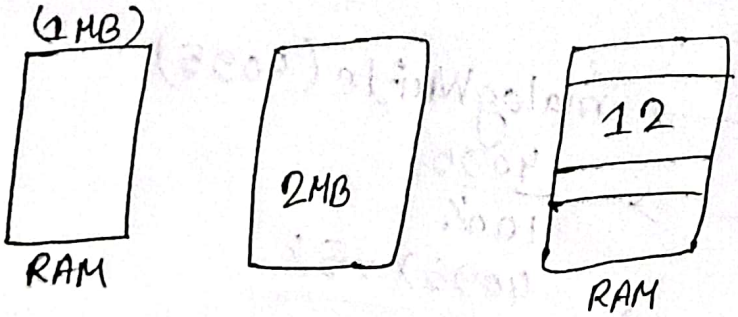
" (1) → $\frac{100\%}{2047}$

" (1535) → $\frac{100\% \times 1535}{2047}$



***10marks

Real Mode:



*Real Mode এর Problem: এতে কোনো protection থাকে না।

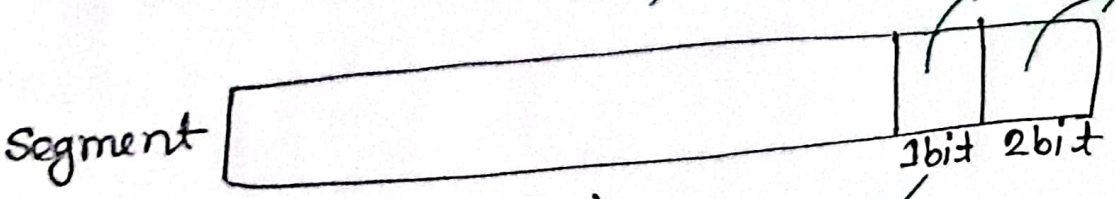
* Virtual Memory:

* Access rights:

0 1
↓ ↓
write Read

* protected Mode Operation (80286):

Segment (16 bit)
↳ Selector



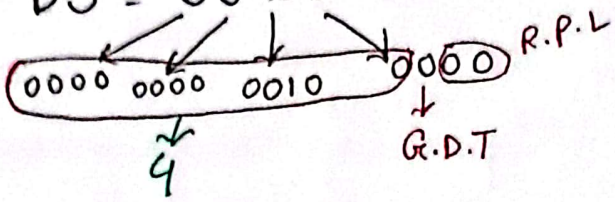
1 bit → GDT/LDT
0 (G.D.T)
1 (L.D.T)

last 2 bit → R.P.L
Requested Privilege Level. (00 is highest and 11 is lowest)

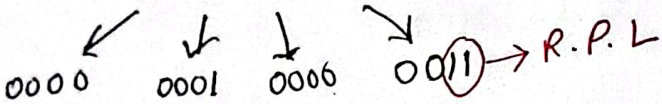
Descriptor table.

*Exercise from Slide:

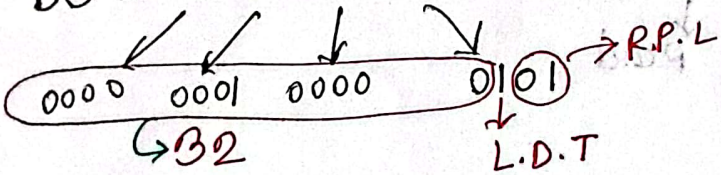
29. DS = 0020H



30. 0103

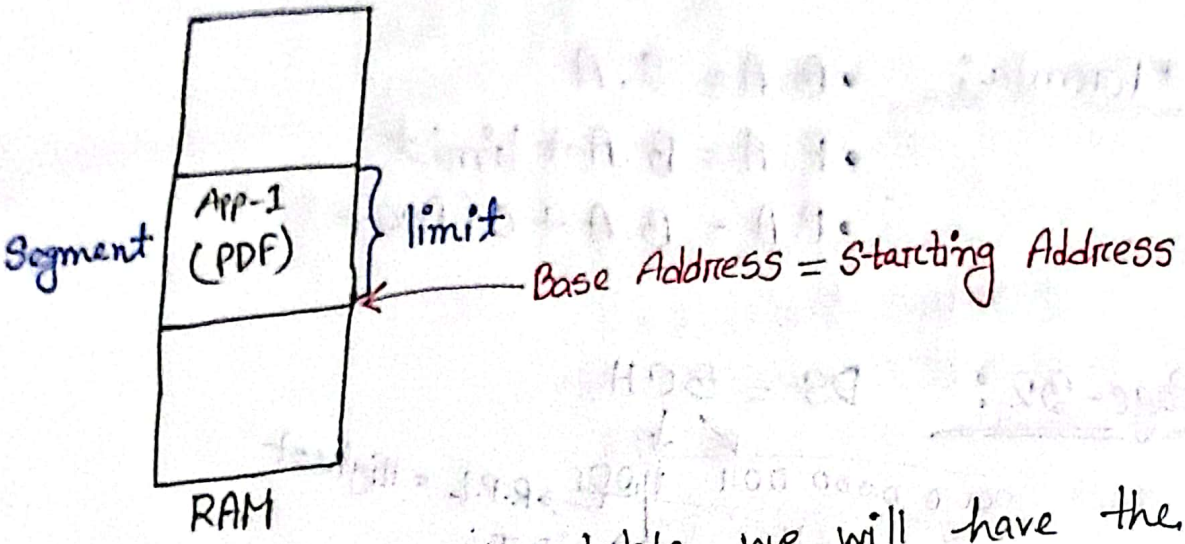


31. DS = 0105



25.01.25

Descriptor Table:



from this description table, we will have the info of

- access
- limit (size)
- Base address (starting)

80286:

B.A = 24 bit

Limit = 16 bit

80386:

B.A = 32 bit

Limit = 20 bit / 32 bit

G = 0 / 1 [G 0 হলে limit 20 bit and 1 হলে 32 bit]

Q: B.A = 10000000H, Limit = 001FFH

G = 0

∴ S.A = B.A = 10000000H

E.A = ? → formula of Ending Address.

∴ E.A = S.A + Limit

= 10000000 + 001FFH

** Limit always 32 bits থাকবে But 20 bit-এ, যদি 32 bits বানাতে হয় সেক্ষেত্রে Limit এর value এর last-এ 3 টা F বসিয়ে 32 bit বানাবো।

*Formula:
- B.A = S.A
- E.A = B.A + limit
- P.A = B.A + offset

*Page-32:

DS = BCH

0000 0000 0011 | 100
Row = 7H
100 → R.P.L = Highest
L.D.T

S.A = B50000H = B.A
E.A = S.A + Limit
= B50000 + OFFFH
= B50FFFH

P.A = B50000 + 10H
=

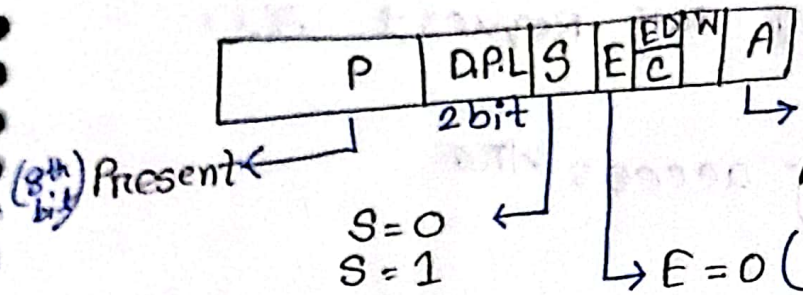
*Page-35:

ACG H

0000 1010 1100 | 0 | 10
158H
10 → R.RL = 10
L.D.T

28.01.25

* Access Right (8-bit) → from description table



- E এর value 0 হলে 3rd bit-এ ED হবে। ED either 0 or 1 হবে
- E এর value 1 হলে 3rd bit এ C বসবে। C = 0 ignore D.P.L
  C = 1 donot ignore D.P.L

- E এর value 0 হলে, W বসবে। here, W = 0 (has not been written)
  write ← W = 1 (has been written (overwrite হবে)

- E এর value 1 হলে, R বসবে। here R = 0 (donot have permission to read)
  Read ← R = 1 (have permission)

Q: Access Right = B4 H

1011 | 0100 → A=0 (can't access)
0100 → ED=1 → Stack
0100 → ED=0 → data/Stack

- S = 0 (System → OS descriptor)
  S = 1 (Application program)

- D.P.L → 6th and 7th bit

- R.P.L → Segment এর জন্য Request করে।

ex: R.P.L = 00, D.P.L = 11 } access পাবে

R.P.L = 11, D.P.L = 00 } access পাবে না

- P(Present) → 8th bit -এ থাকে
  P = 0 (undefined)
  P = 1 (defined)

Q: 500H → S.A = B56700H
E.A = B56829H

B56829H ↕ B56700H] B123ABH

242
Q: D1 → 0001 → A=1 (can't access)
↓ ED=0 → Data
ED=0 → Data

10
P=1 DPL=10

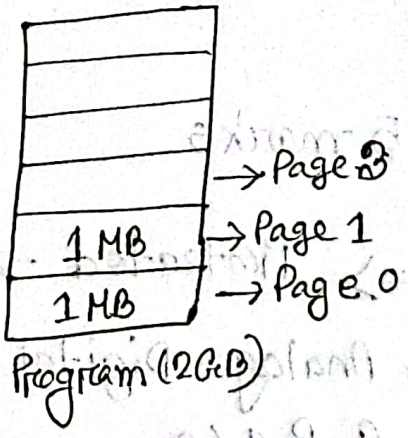
* Spring - 24:

a) 28 H → 0000 0000 0010 1000 (5H)

R.P.L = Highest

G.D.T :

Paging: (For theory)



* **Page Fault:** কোনো Page খুঁজতে গিয়ে যদি RAM-এ Page টি পাওয়া না যায়, তখন Page fault হয়।

Final Syllabus

1. Bus Timing → Diagram
2. pipelining → Diagram
3. SAP → 10 marks
4. Communication Protocol - 1 → T
5. " " - 2 → Diagram
6. microcontroller note
7. analog - digital →
8. Protected mode → 10 marks
9. Paging

* Theory ⇒ 3 - 5 marks

** Marks → Q-1 ⇒ Protected → 10

Q-2 ⇒ Analog - Digital
C.P-1/2
Microcontroller note
} → 10

Q-3 ⇒ Bus timing
pipelining
Pagine
} → 10

Q-4 ⇒ SAP → 10 marks

CamScanner