MICROPROCESSORS

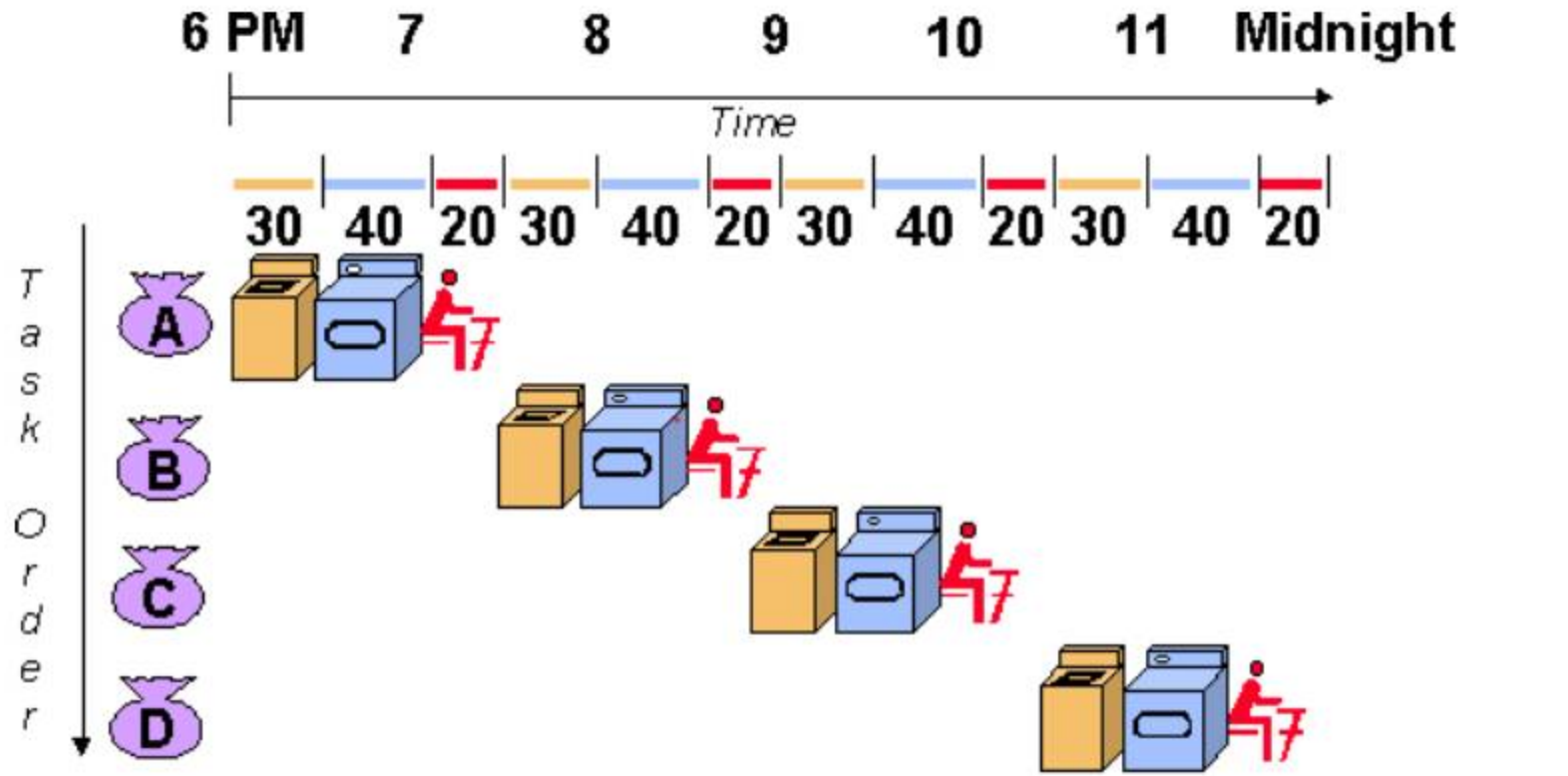
8086 Architecture and memory segmentation

PROF POONAM PANGARKAR, ASSISTANT PROFESSOR , CSE-DATA SCIENCE ,APST THANE

Pipelining and parallel processor

An **instruction pipeline** is a technique used in the design of microprocessors to increase the number of instructions that can be executed in a unit of time. Pipeline technique is used in advanced microprocessors where the microprocessor begins operation on next instruction before it has completed operation on the previous. That is, several instructions are simultaneously in the *pipeline* at a different stage of processing.

The pipeline is divided into different Stages and each Stage can perform its particular operation simultaneously with the other stages. When a stage completes an operation, it passes the result to the next stage in the pipeline and fetches the next operation from the preceding stage. The final results of each instruction emerge at the end of the pipeline in rapid succession. Since all units perform operation concurrently on different instructions , it is known as parallel processor.

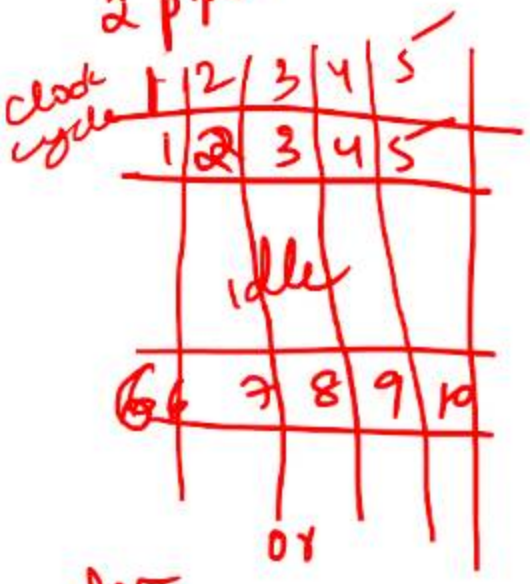


Source: http://www.ece.arizona.edu/~ece462/Lec03-pipe/

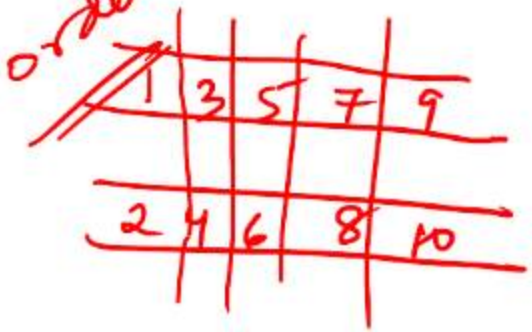
P1

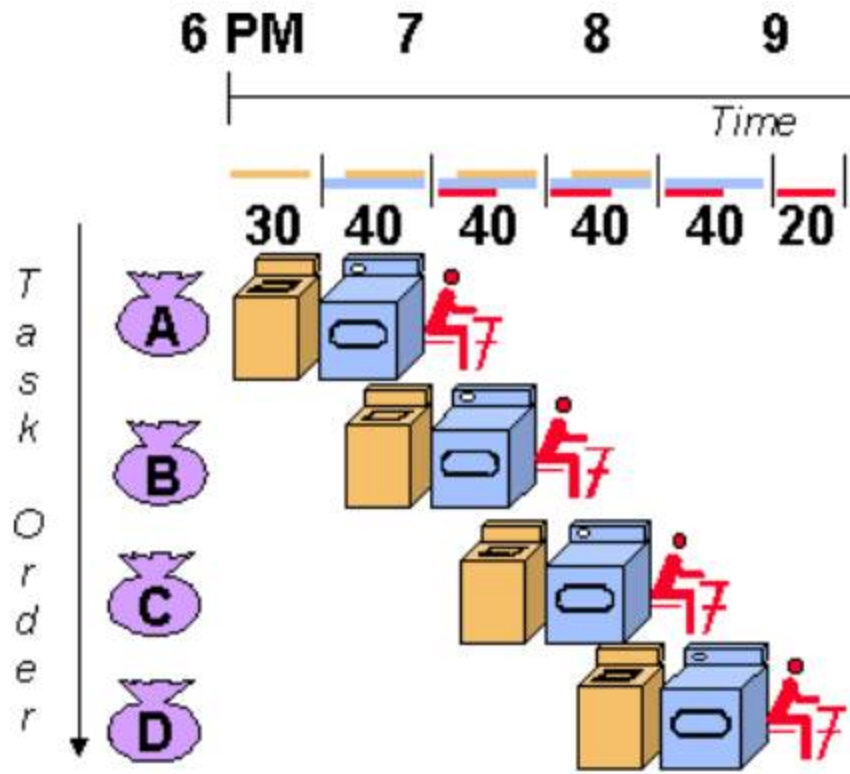
5 stage pipelining

2 pipes

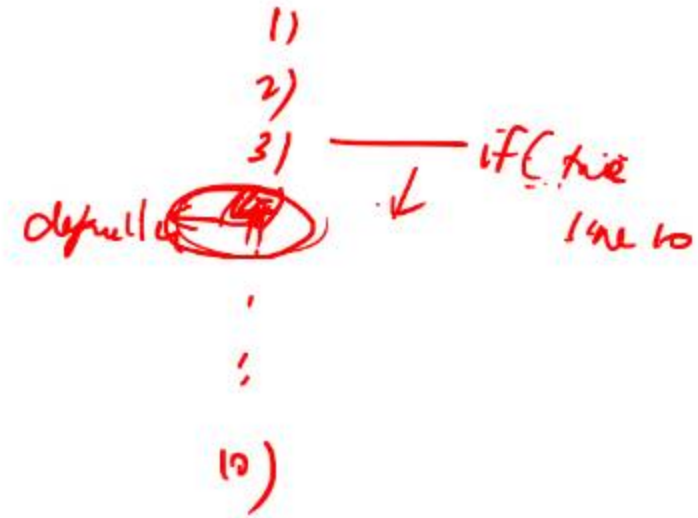


order





branch instructions



Pipelining and parallel processor

Pipelining of 8086

INSTRUCTION NO.	EXECUTION PHASES							
1	Fetch 1	Decode-1	Execute-1					
2		Fetch-2	Decode-2	Execute-2				
3			Fetch-3	Decode-3	Execute-3			
4				Fetch-4	Decode-4	Execute-4		
5					Fetch-5	Decode-5	Execute-5	
6						Fetch-6	Decode-6	Execute-6
Machine cycle	1	2	3	4	5	6	7	8

3 stage
fetch
decode
execute

Non-Pipelining Process of 8085

	Inst ruction-1			Inst ruction 2			Ins' ruction-3		
	Fetch-1	Deco e-1	Execute-1	Fetch-2	Decode-2	Execute-2	Fetch-3	Decode-3	Execute-3
M. cycle	1	2	3	4	5	6	7	8	9

Network
P4 → 80.

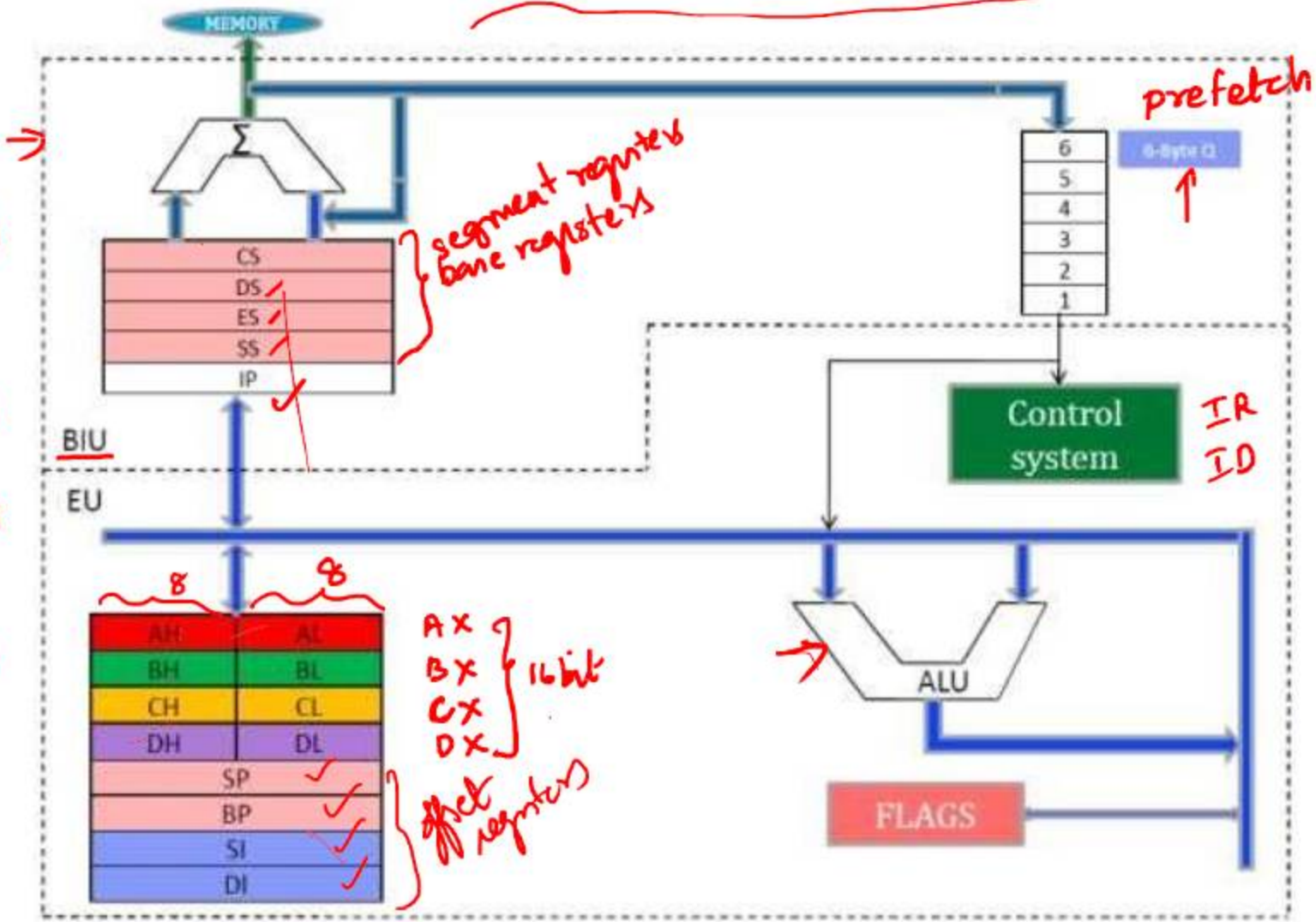
4 pipe
↓
20 stages

Superscalar pipelining
↓
2 pipes
↓
5 stages
↓
10 instructions

Architecture of 8086

- The architecture of 8086 provides a number of improvements over 8085 architecture. It supports a 16-bit ALU, a set of 16-bit registers and provides segmented memory addressing capability, a rich instruction set, powerful interrupt structure, fetched instruction queue for overlapped fetching and execution.
- The complete architecture of 8086 can be logically divided into two units a) Bus Interface Unit (BIU) and (b) Execution Unit (EU).
- Both units operate asynchronously to provide the 8086 to overlap instruction fetch and execution operation , which is called as parallel processing. This results in efficient use of the system bus and enhance system performance.

Architecture of 8086



physical address calculation

20 bits

PA = Segment Address x 10H + offset address

PA = SA x 10H + OA

memory segmentation

ADD AL, BL

seg 7 of 3 special purpose

1234

AH AL

ADD source destination

AL ← AL + BL

04H 05H

Memory Segmentation

Code / Data / Stack

100 2byte
102
104
:

Data ↑ ↓

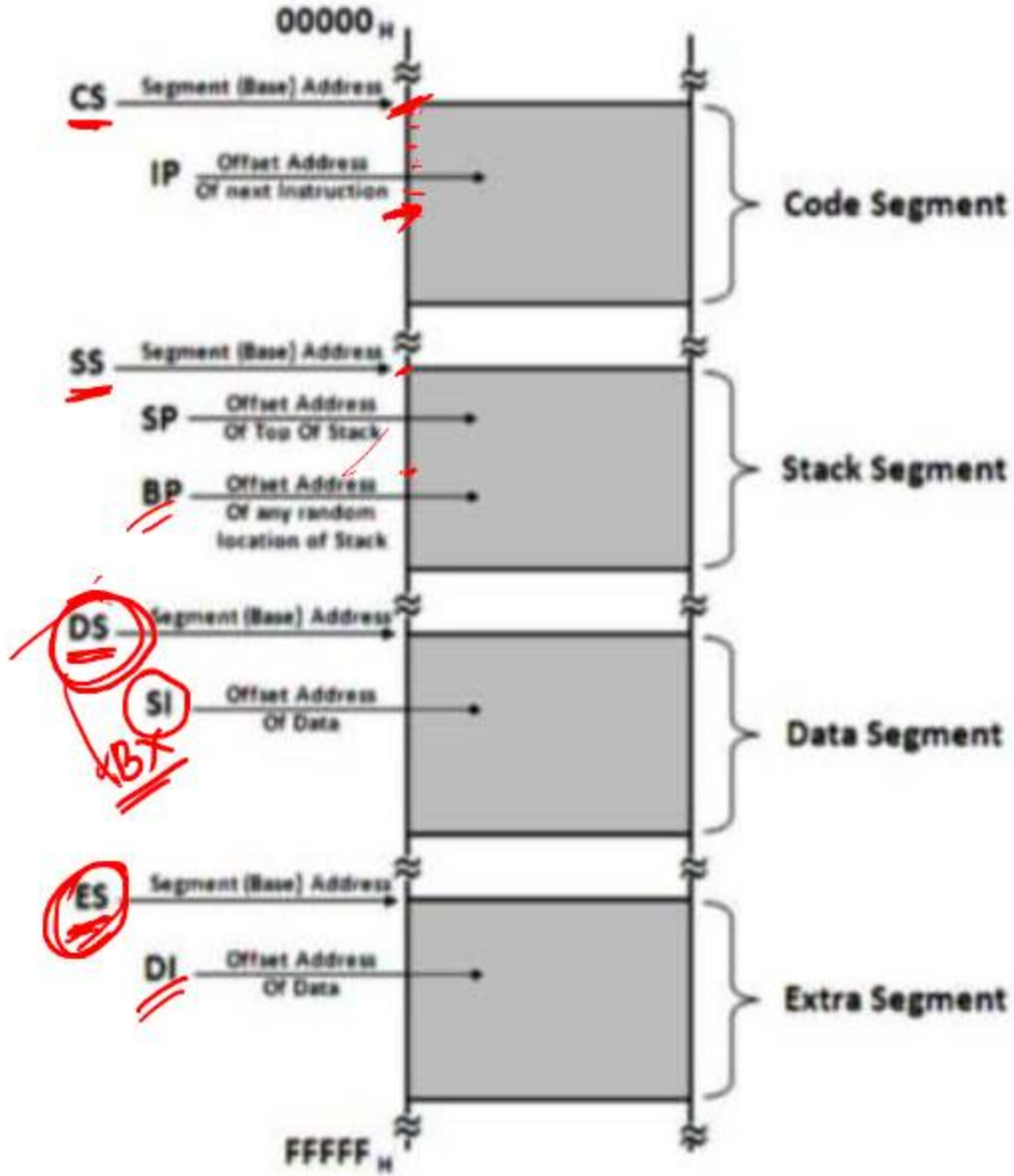
20 bit

bytes

1 byte = 8bits

16bits

3 mly



8085 ⇄ 8 bits
16 bit

8086 ⇒ 16 bit
20 bit

Virtual mly

PA = SA (10H) + OA

20 bits

PA = 1234 × 10 + 0005
= 12345

SA + OA

1234
0005

PA = 1234 × 10 + 0005

0000

20 bits

1 2 3 4

0001 0010 0011 0100 × 10^{H} + 0A

SA 0000 0000 0000 0101

12345

Max → 64 KB

Minimum → 10 H

1MB 16 segments

PA = SA × 10H + OA

12340 23750

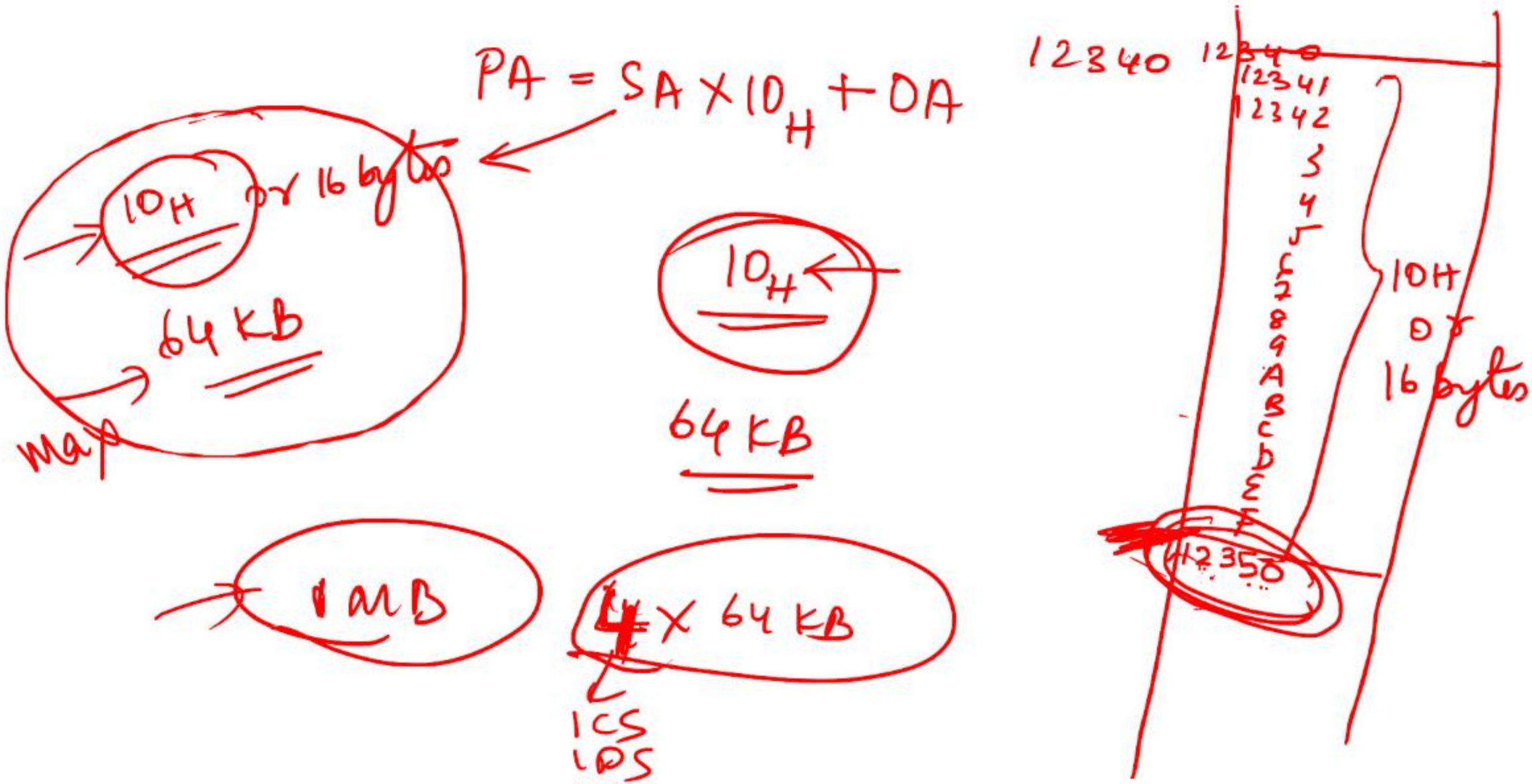
12345

12340

12340 10H

12350

1 2 3 4 5 6 7 8 9 A B C D E F 10


PA = SA × 10H + OA
10H or 16 bytes
64 KB
map
10H
64 KB
1 MB
4 × 64 KB
ICS
IDS
IES
ISS
12340
12340
12341
12342
10H
or
16 bytes
12350

1 MB

16 bit

16 bit

20 bit

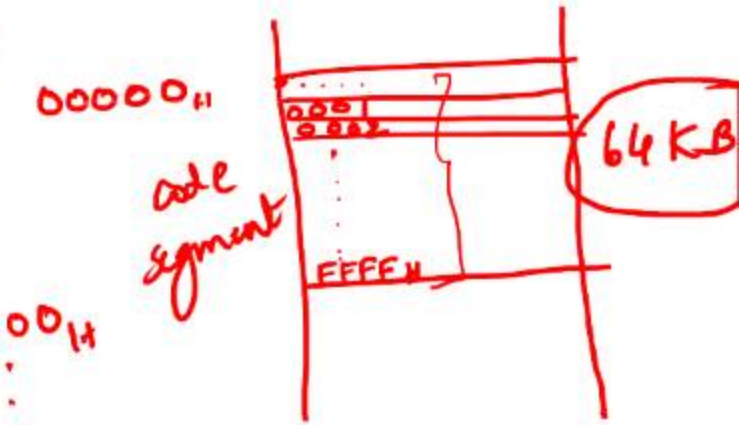
Seg Addr

$\times 10_H$ + offset

PA

00000_H

$FFFFF_H$



12340

12350

minimum size

10H or 16 bytes

12345

12341H $\times$ 10H + 0005H

16 bit

0000_H … $FFFF_H$

16 bit

10_H

10H = 16 bytes

12340 → 4F → 12350

12341

42

43

49

4A

4B

4C

4D

4E

0001 0010 0011 0100 $\times$ 10_H

0001 0010 0011 0100 0000 +

0000 0000 0000 0101

0001 0010 0011 0100 0101

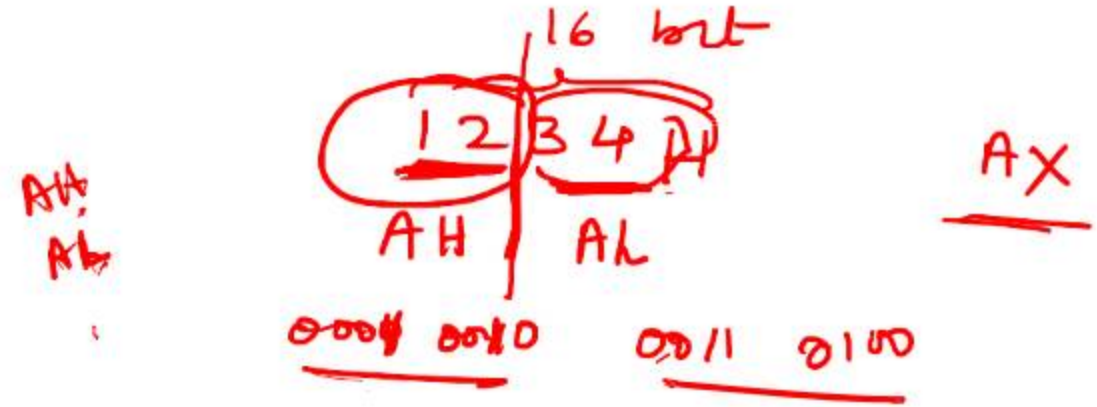
1 2 3 4 5

If CS = 3000

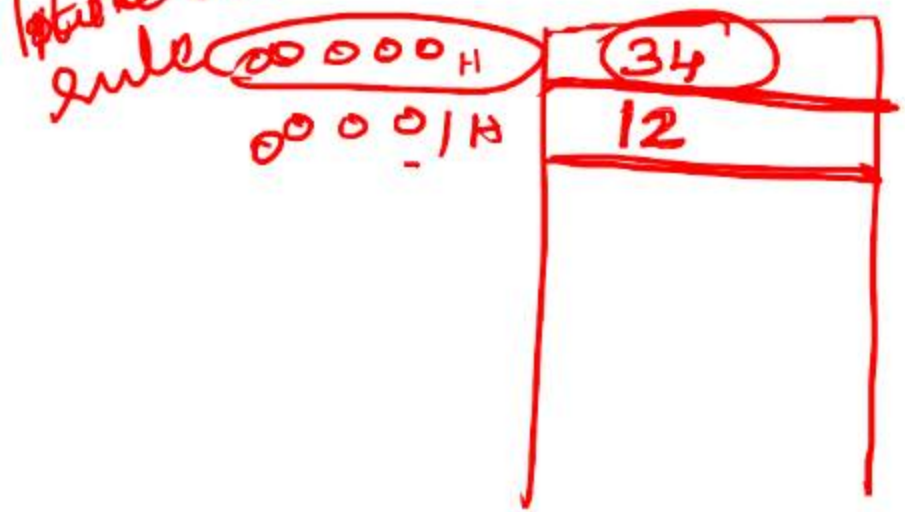
$PA = 3000 \times 10_H + 0000_H$

$= 30000$

PA = Seg Addr × 10_H + offset address.

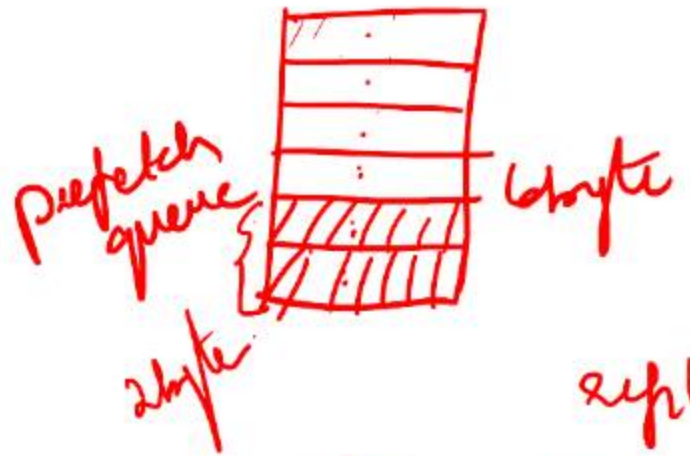


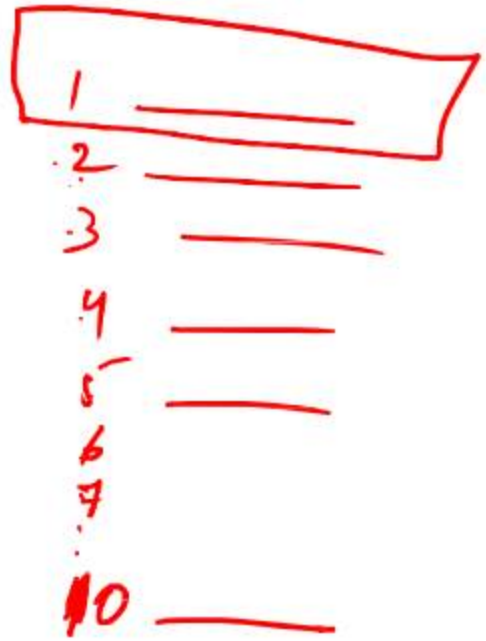
little endian rule



The size of the prefetch queue is 6 bytes.

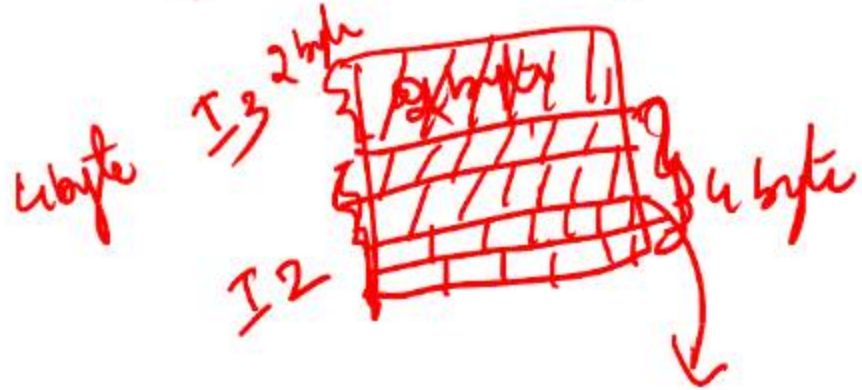
of the program

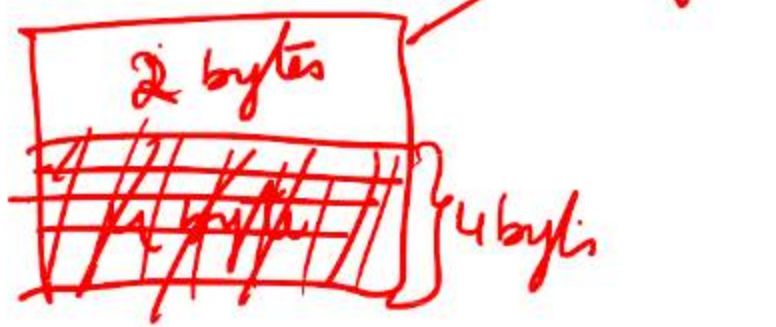




1 byte
2 byte
3 bytes

refilled?






opcode
operand
MOV AL, BL
ADD BL, AL
MOV AL
25H
MOV AX
2000H