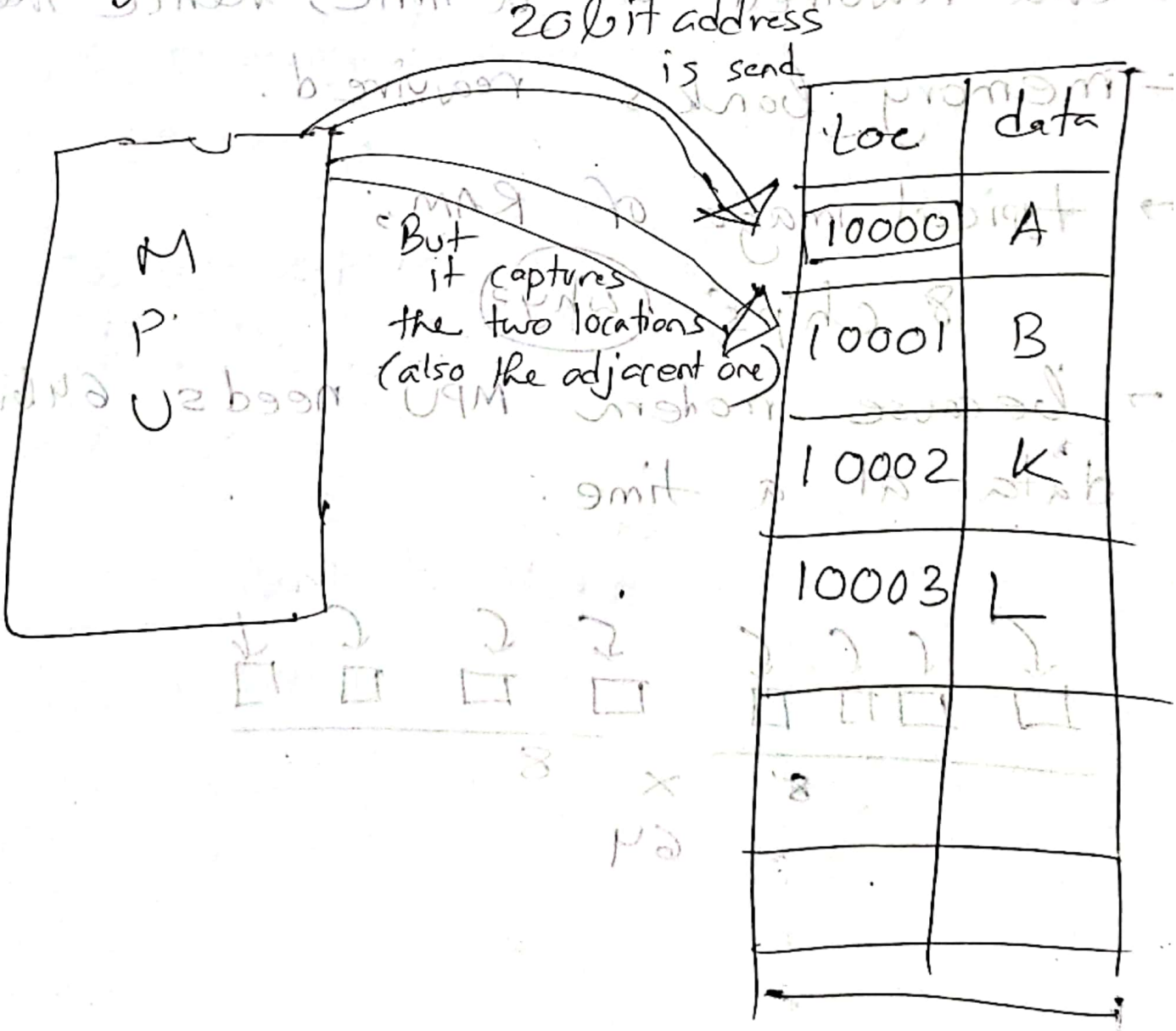
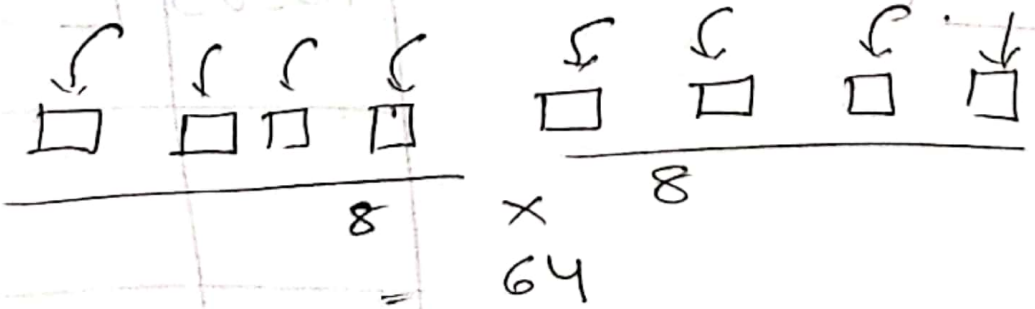
8086 Memory Banking

→ has two chips / two banks
→ one has even addresses (even ban
→ " " odd " " (odd ")
→ just like your any book.
→ alligned data & misalligned data?

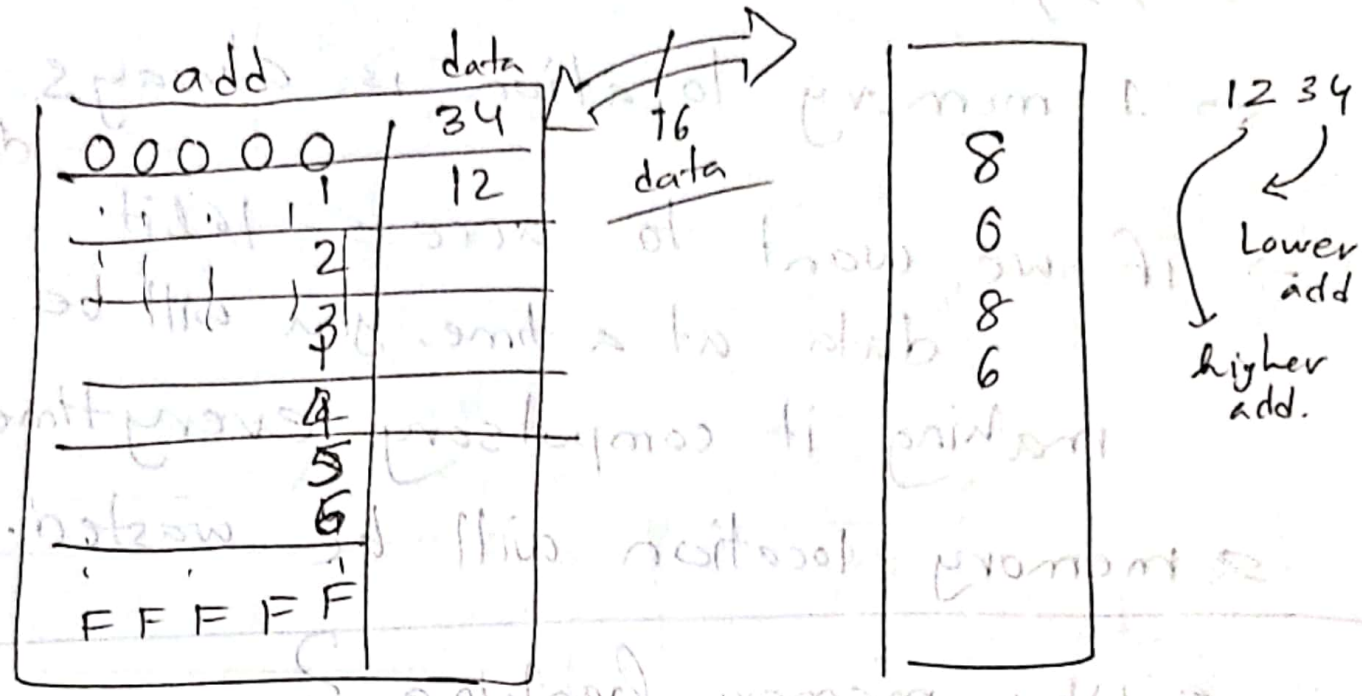


the reason why?

→ all MPU after 8086, have this feature
→ all modern processor has 8 banks.
→ 8085 did not have banking
~~→ 80386 accesses~~
→ comes from the concept of size of the data bus; ~~da~~
→ data reavired at a time; hence the ore memory banks reavired.
→ typical image of RAM;
8 chips (why?)
→ because modern MPU needs 64bits data at a time.



1] Why every location cannot be 32 bit?



⇒ 1 memory location can carry 1 byte = 8 bits data

→ So, µp can generates one address at a time; but µp wants to access 16 bit data at one cycle.

→ ~~3rd~~ you want to denomination has to be as small as possible.

→ if everyone has 1.000 taka. either the shopkeeper or the buyer has to loose the money.

→ 95% we work on bytes.

↔ 1 memory location is always 8bit data.

→ if we want to access 16 bit data at a time, you will be making it compulsory everytime, so, memory location will be wasted.

→ Why memory banking?

because we
⇒ do not want to waste memory location

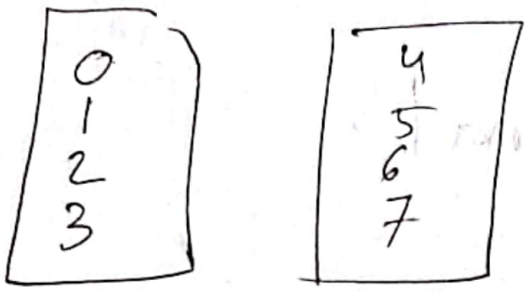
⇒ do not want to make it compulsory to make 16-bit data transfer at a time.

→ 16-bit data transfer can happen at one cycle too

⇒ All in all

→ i can use one hand

⇒ i can use both hands whenever required.



= wrong concept.

Consider like a ~~2440~~ book.

the only bit different is A_0

A_{19} - - - - - A_4	A_3	A_2	A_1	A_0	dec
	0	0	0	0	0
	0	0	0	1	1
	0	0	1	0	2
	0	0	1	1	3
	0	1	0	0	4
	0	1	0	1	5
	0	1	1	0	6
	0	1	1	1	7
	1	0	0	0	8
	1	0	0	1	9

two different pattern, MP can never generate two pattern at one time

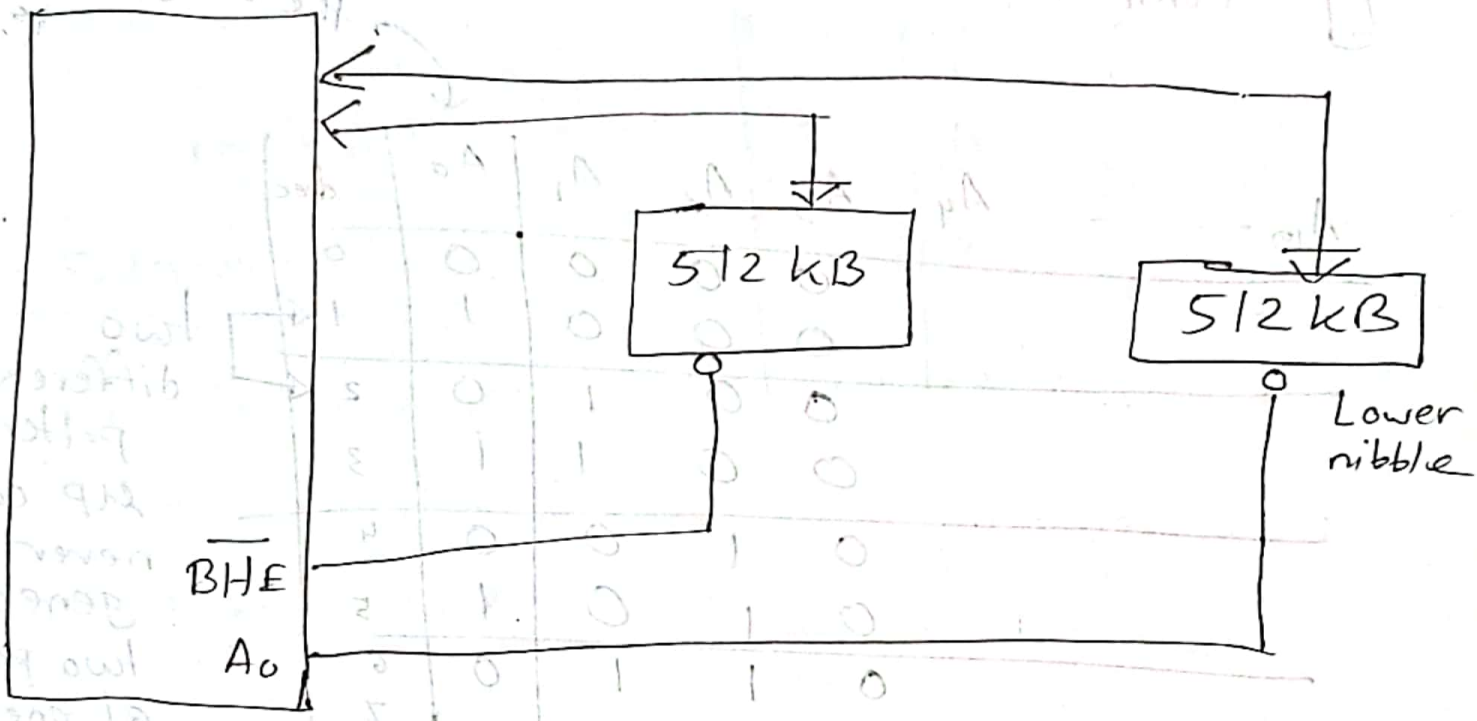
Chip - 1

A_2	A_1	A_0
0	0	1
0	1	1
1	0	1
1	1	1

Chip - 0

A_2	A_1	A_0
0	0	0
0	1	0
1	0	0
1	1	0

1) $\overline{CS}$ → chip select
Active low signal.

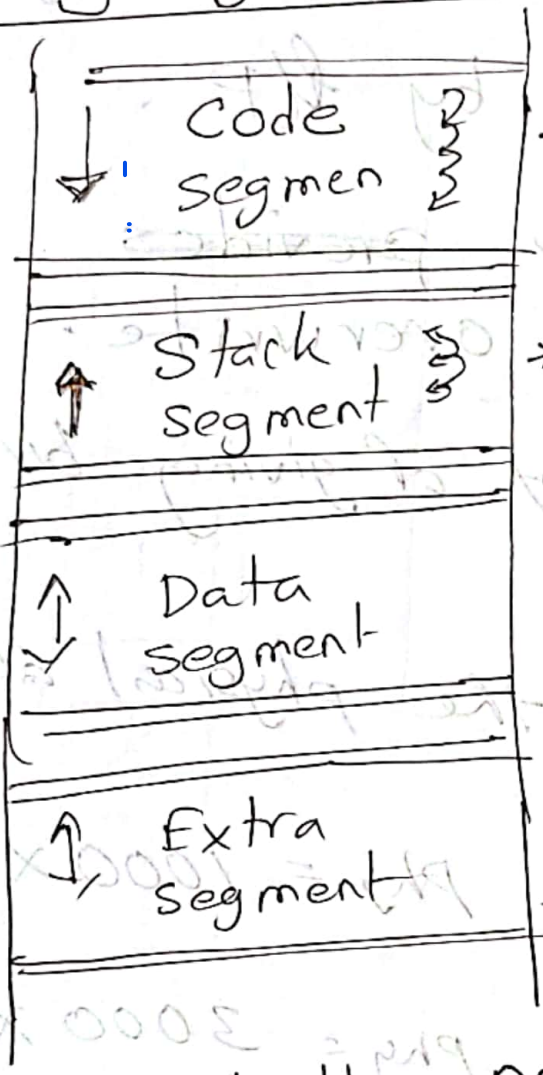


A_0 = is only for the lower bank.

2) For higher bank, up produces bus high enable signal. (bank high enable) (BHE)

$\overline{BHE}$	A_0	operation
0	0	R/W 16 bit from both banks
0	1	R/W 8 bit from the higher bank.
1	0	R/W 8 bit from the lower bank.
1	1	None of the operations will be selected. (IDLE)

II Memory Segmentation



→ Programs will go downwards (incremented IP).

→ structured forms of data. (sms folder) LIFO / go upwards.

→ Random data you can store any where, can go downwards or upwards.

* But eventually no one remembers what is stored in 1 MB memory locations. (1 million)

⇒ How they are virtually divided?

By declaring physical address range during PC assembling

→ Segments are created by programmer

→ but managed by OS.

→ Segmentation ~~provides~~ prohibites overwrite.

→ this is the concept of giving birth of file/folders.

→ MP calculates the physical address.

CS = 1000

SS = 3000

DS = 5000

Phy = 1000 × 10

Phy = 3000 × 10

Phy = 5000 × 10

50000

5FFFF

10H = 16