8086 Memory Banks

Dept. of Computer Science and Engineering
BRAC University
CSE 341 Team

Lecture References:

- **Book:**
  - *Microprocessors and Interfacing: Programming and Hardware,*

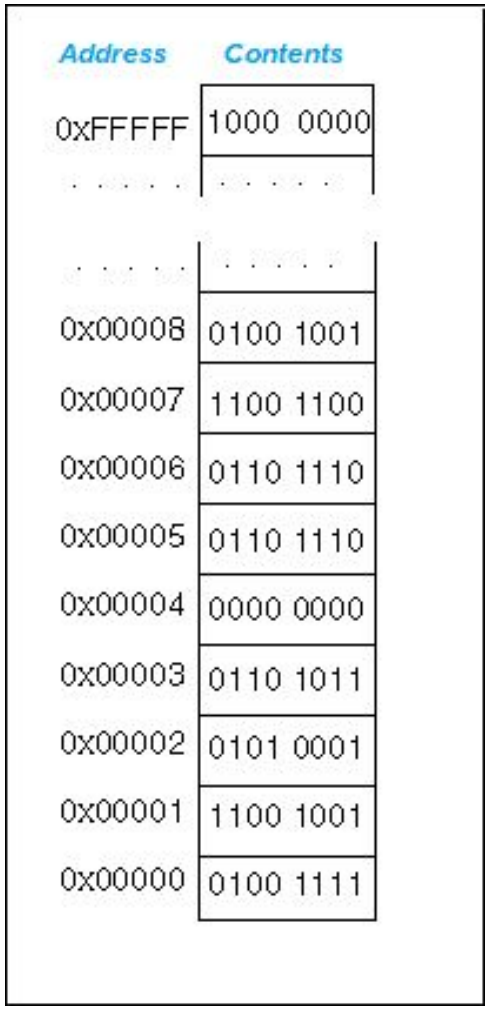
    Author: Douglas V. Hall
  - *The 8086/8088 Family: Design, Programming, And Interfacing,*

    Author: John Uffenbeck.

8086 Memory Organisation

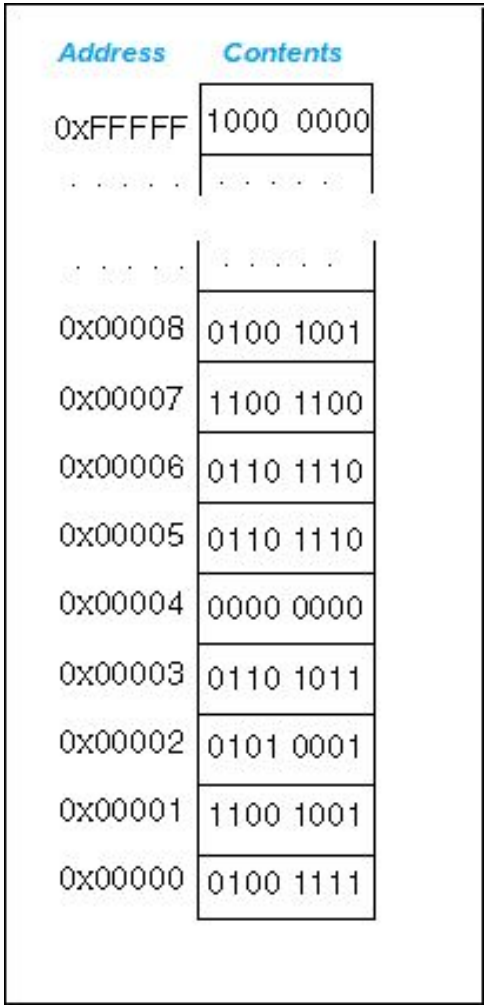
? **RECAP:**

- The 8086 has 20-bit address bus, so it can address 2^{20} or 1,048,576 addresses.
- Each address can store a byte. Hence, 8086 can store upto 1MB.
- Each read/write operation takes 1 bus cycle



8086 Memory Organisation

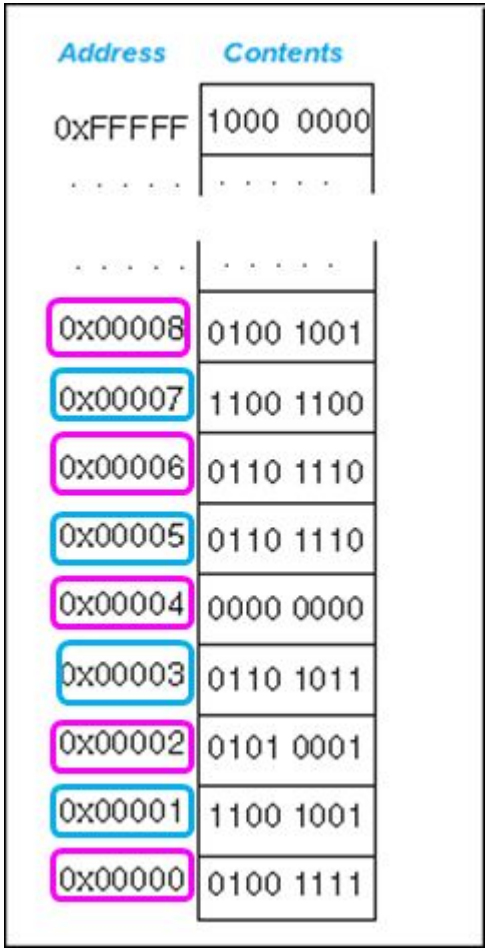
- Each read/write operation takes 1 bus cycle for 1 byte data
- To read/write 1 word or 2 bytes of data, the µp would need 2 cycles
- To solve this problem by saving processing time, memory is organized into *memory banks*
- Odd and even banks
- With the use of A_0 and $\overline{BHE}$ pins



8086 Memory Organisation

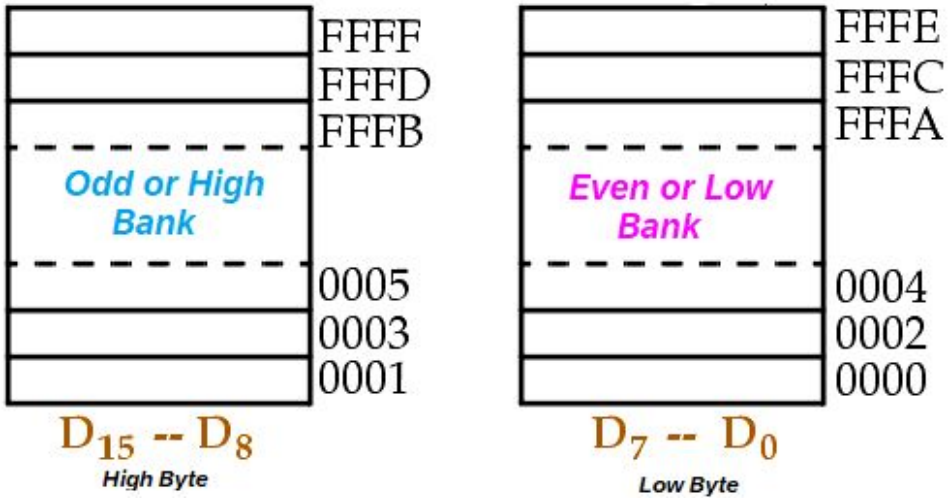
- Addresses are consecutively numbered
- All even numbers end with a 0 and all odd numbers end with a 1

Even		Odd	
6 =	110	5 =	101
14 =	1110	13 =	1101
20 =	10100	27 =	11011
42 =	101010	35 =	100011



8086 Memory Organisation

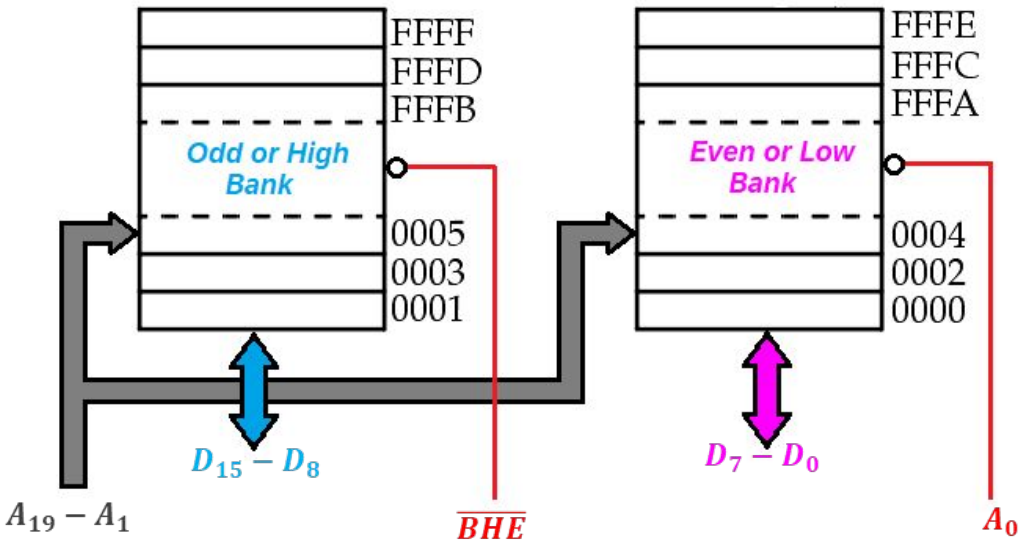
- Even addresses are considered as the even/low bank, which holds the content of the low byte while
- Odd addresses are considered as the odd/high bank, which holds the content of the high byte



8086 Memory Banks

8086 Pin Recap:

- $A_0 - A_{19}$ pins for carrying 20-bit address
- $D_0 - D_7$ pins for carrying low data byte and $D_8 - D_{15}$ for high byte
- $\overline{BHE}$ when 0 indicates data bus carries high byte of data



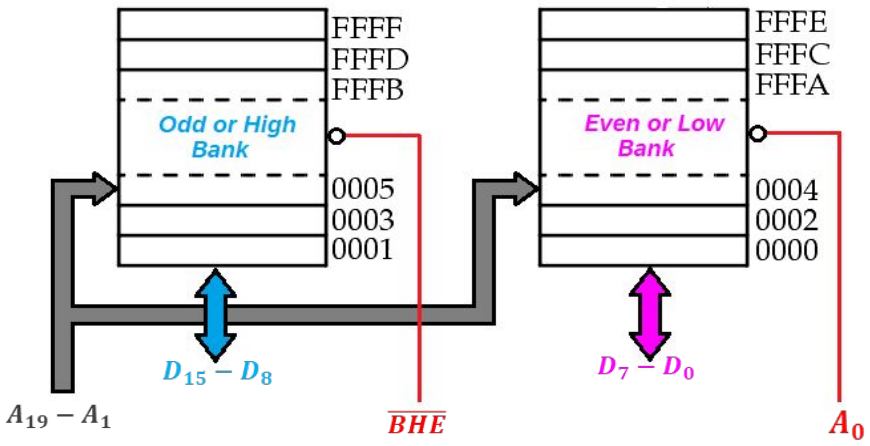
- The memory for an 8086 is set up in to 2 banks of up to 524,288 bytes or 512kB each
- This makes it possible to read/write a word with one machine cycle

8086 Memory Addressing

Data can be accessed from the memory in 4 different ways:

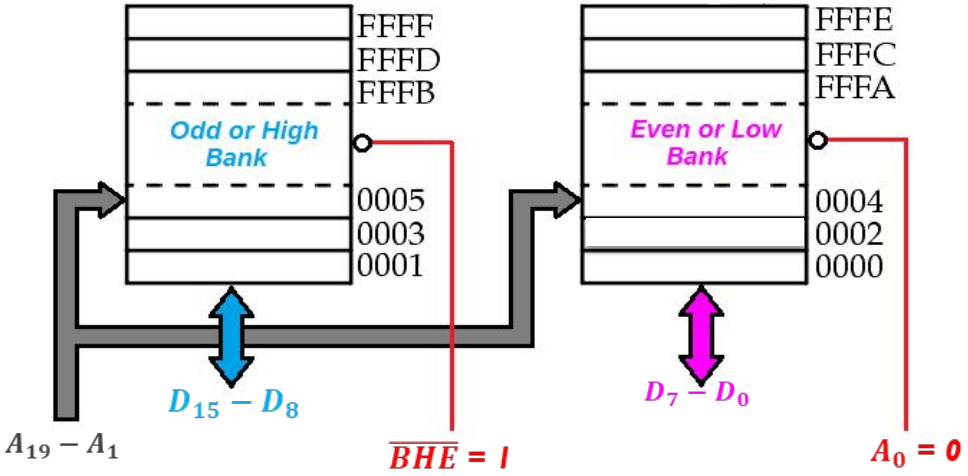
- 8 - bit data from Even Bank e.g. from address 00002
- 8 - bit data from Odd Bank e.g. from address 00003
- 16 - bit data starting from Even Address e.g. from 00002 and 00003
- 16 - bit data starting from Odd Address. e.g. from 00003 and 00004

		Type of Transfer	Data Lines
0	0	Word i.e. a byte from each bank	
0	1	Byte from odd bank	
1	0	Byte from even bank	
1	1	None	-



A byte from Low/Even bank

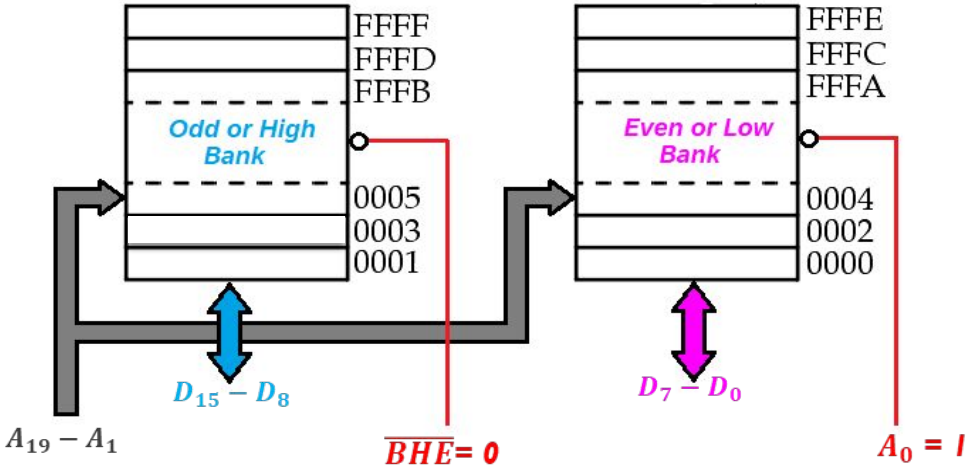
- Requires one bus-cycle to read/write a data-byte.
- Valid address is provided via $A_{19} - A_1$ with $A_0 = 0$ & $\overline{BHE} = 1$
- Byte of data fetched on $D_7 - D_0$
- Low bank ***enabled***, High bank disabled



		Type of Transfer	Data Lines
1	0	Byte from even bank	

A byte from High/Odd bank

- Requires one bus-cycle to read/write a data-byte.
- Valid address is provided via $A_{19} - A_1$ with A_0= 1 & $\overline{BHE}$ =0
- Byte of data fetched on $\boldsymbol{D_{15} - D_8}$
- Low bank disabled, High bank ***enabled***

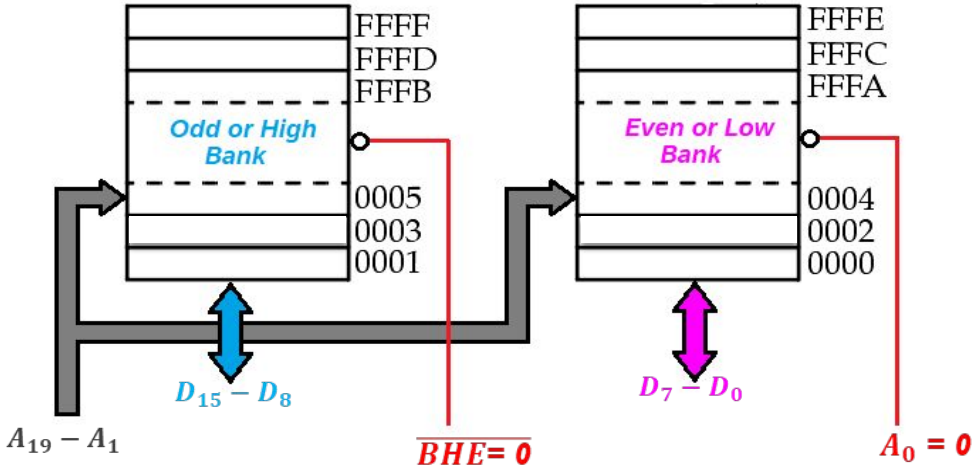


		Type of Transfer	Data Lines
0	1	Byte from odd bank	

An aligned word

*starting from an **even** address*

- Requires one bus-cycle to read/write a data-word.
- Valid address is provided via $A_{19} - A_1$ with $A_0 = 0$ & $\overline{BHE} = 0$
- Word of data fetched on $D_{15} - D_0$
- Low bank **and** High bank ***enabled***

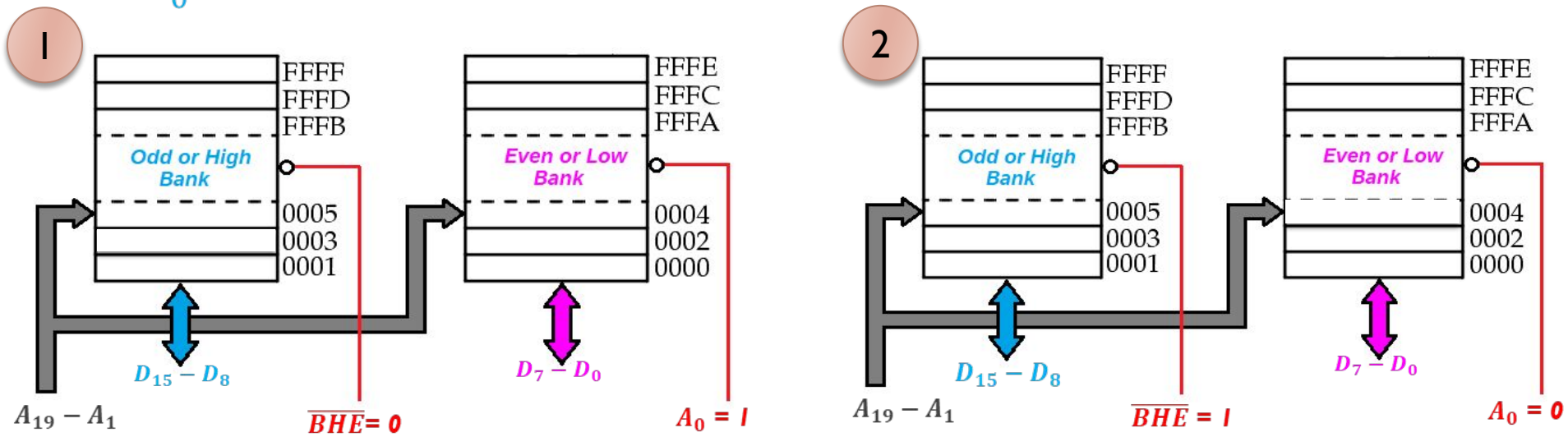


		Type of Transfer	Data Lines
0	0	Word i.e. a byte from each bank	

An unaligned word

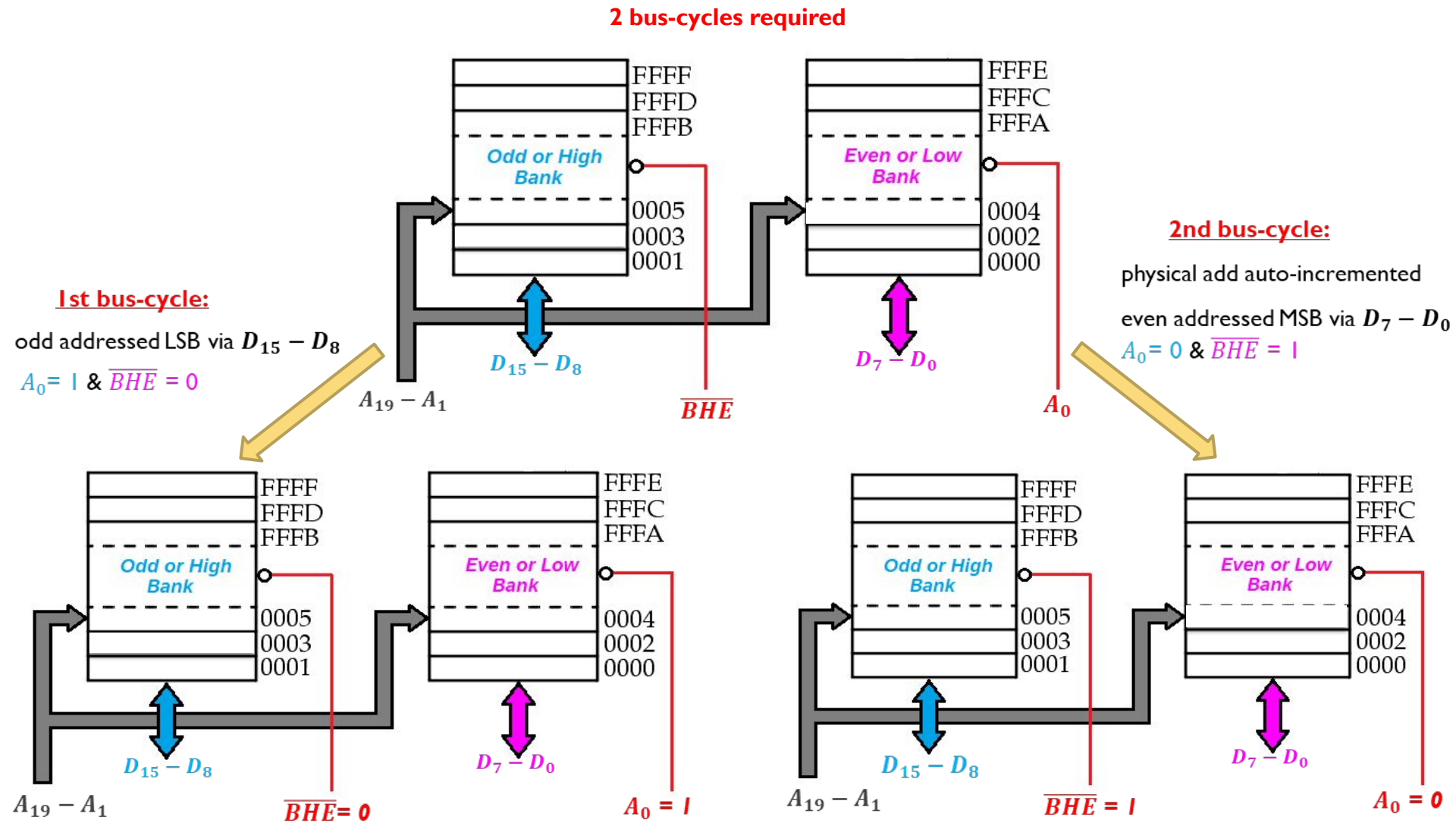
*starting from an **odd** address*

- Requires **two bus-cycles** to read/write a data-word
- **During the 1st bus-cycle**, odd addressed LSB of the word is accessed from the high memory bank via $D_{15} - D_8$ of data bus; A_0= 1 & $\overline{BHE}$ = 0
- **During 2nd bus-cycle**, physical address is auto-incremented to access the even address MSB of the word from the Low bank via $D_7 - D_0$; A_0= 0 & $\overline{BHE}$ = 1



An unaligned word

*starting from an **odd** address*



BRAC UNIVERSITY
Inspiring Excellence

Thank You