# 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

| Address | Contents |
|---------|-----------|
| 0xFFFFF | 1000 0000 |
| . . . . . . | . . . . . . |
| . . . . . . | . . . . . |
| 0x00008 | 0100 1001 |
| 0x00007 | 1100 1100 |
| 0x00006 | 0110 1110 |
| 0x00005 | 0110 1110 |
| 0x00004 | 0000 0000 |
| 0x00003 | 0110 1011 |
| 0x00002 | 0101 0001 |
| 0x00001 | 1100 1001 |
| 0x00000 | 0100 1111 |

# 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

| Address | Contents |
|---------|----------|
| 0xFFFFF | 1000 0000 |
| . . . . . | . . . . . |
| . . . . . | . . . . . |
| 0x00008 | 0100 1001 |
| 0x00007 | 1100 1100 |
| 0x00006 | 0110 1110 |
| 0x00005 | 0110 1110 |
| 0x00004 | 0000 0000 |
| 0x00003 | 0110 1011 |
| 0x00002 | 0101 0001 |
| 0x00001 | 1100 1001 |
| 0x00000 | 0100 1111 |

# 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 = | 11**0** | 5 = | 10**1** |
| 14 = | 111**0** | 13 = | 110**1** |
| 20 = | 1010**0** | 27 = | 1101**1** |
| 42 = | 10101**0** | 35 = | 10001**1** |

| Address | Contents |
|---|---|
| 0xFFFFF | 1000 0000 |
| . . . . . | . . . . . |
| . . . . . | . . . . . |
| 0x00008 | 0100 1001 |
| 0x00007 | 1100 1100 |
| 0x00006 | 0110 1110 |
| 0x00005 | 0110 1110 |
| 0x00004 | 0000 0000 |
| 0x00003 | 0110 1011 |
| 0x00002 | 0101 0001 |
| 0x00001 | 1100 1001 |
| 0x00000 | 0100 1111 |

*even*   *odd*

# 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**

| | FFFF | | FFFE |
|---|---|---|---|
| | FFFD | | FFFC |
| | FFFB | | FFFA |
| *Odd or High Bank* | | *Even or Low Bank* | |
| | 0005 | | 0004 |
| | 0003 | | 0002 |
| | 0001 | | 0000 |

$D_{15} -- D_8$
**High Byte**

$D_7 -- D_0$
**Low Byte**
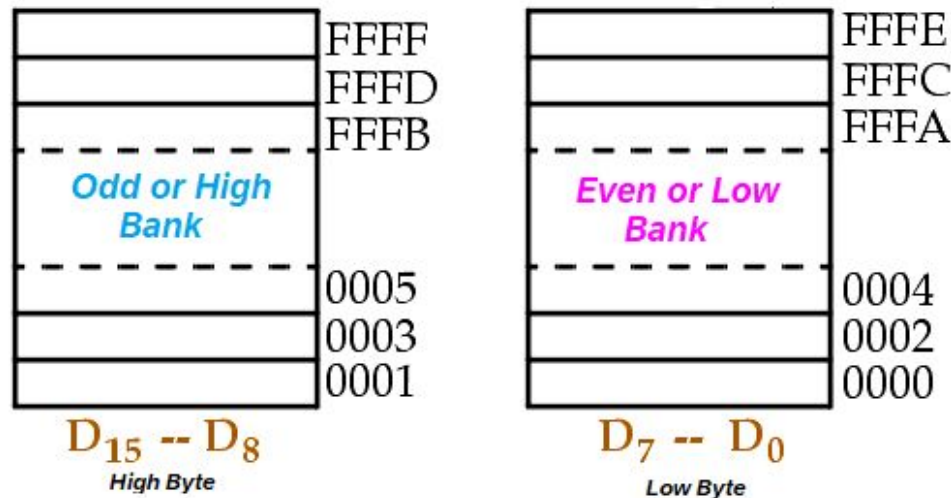
# 8086 Memory Banks

## 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
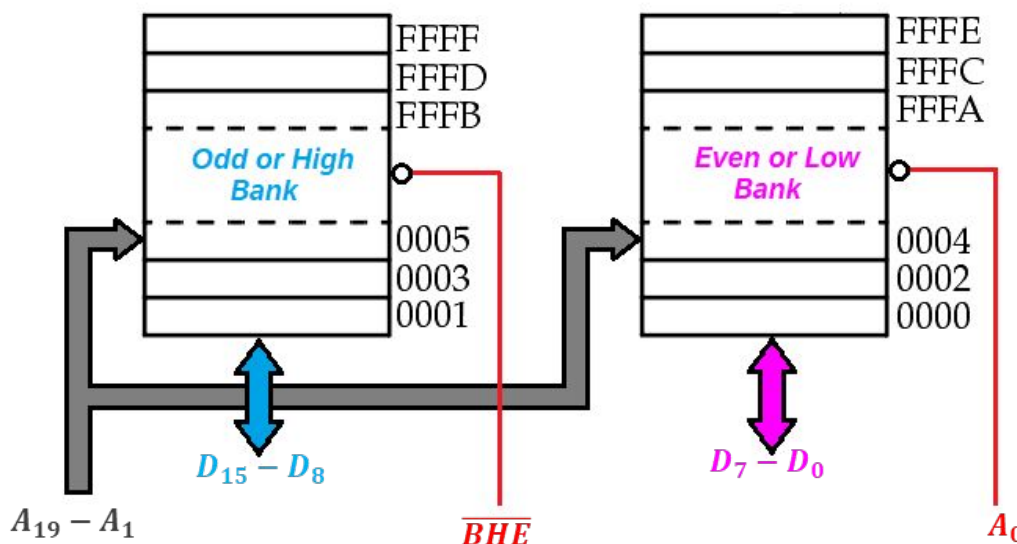


□ The memory for an 8086 is set up in to 2 banks of up to 524,288 bytes or 512kB each

□ 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 | - |



**CSE – 341: Microprocessors**
**BRAC University**
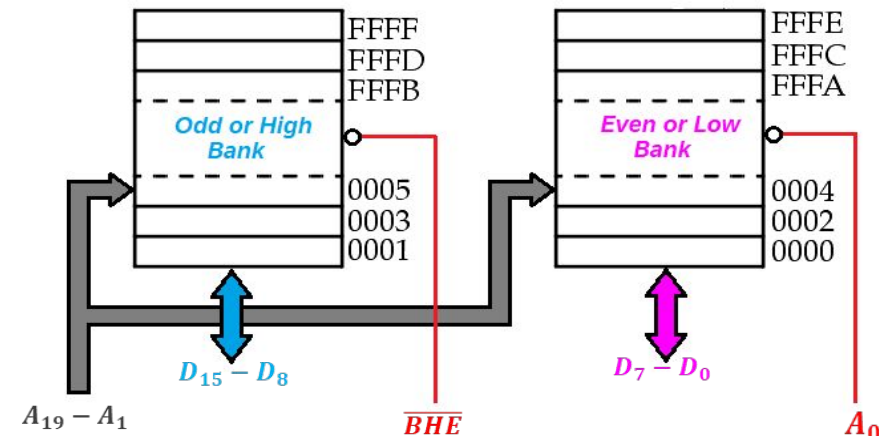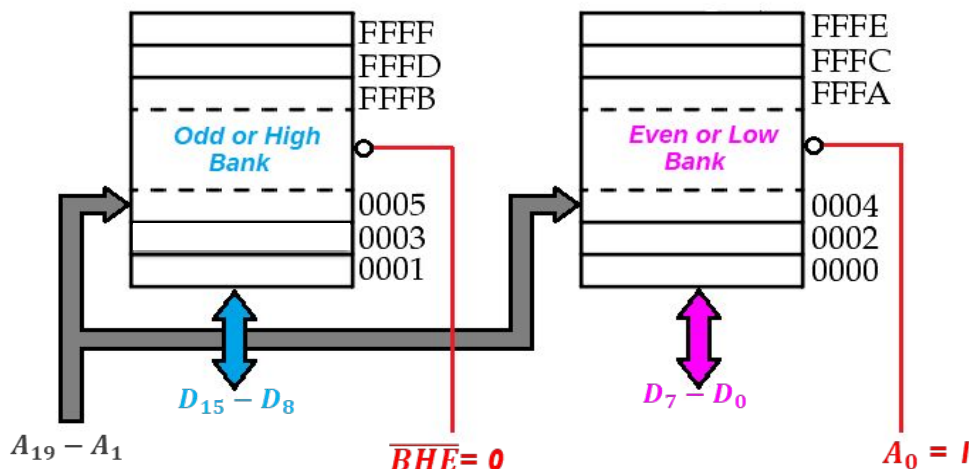
# A byte from Low/Even bank



Inspiring Excellence

- 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
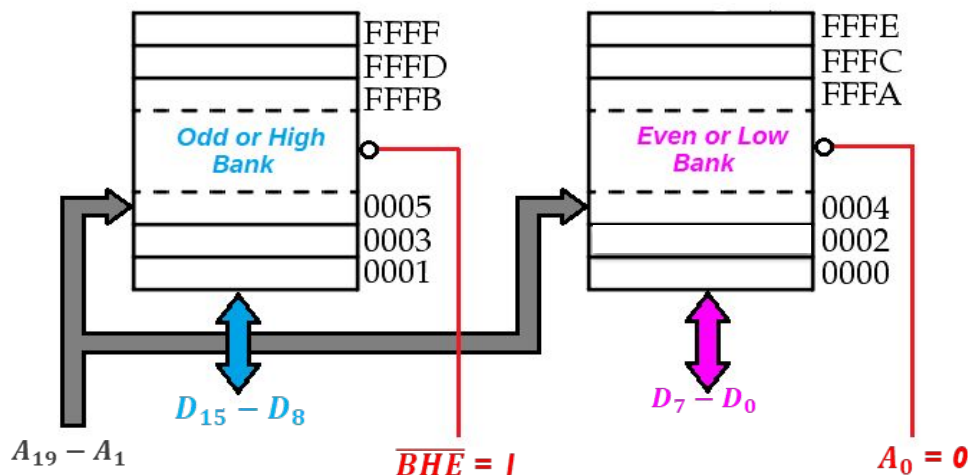


| | | Type of Transfer | Data Lines |
|---|---|---|---|
| 1 | 0 | Byte from even bank | |

$D_{15} - D_8$   $D_7 - D_0$

$A_{19} - A_1$   $\overline{BHE} = 1$   $A_0 = 0$

# 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 $D_{15} - D_8$
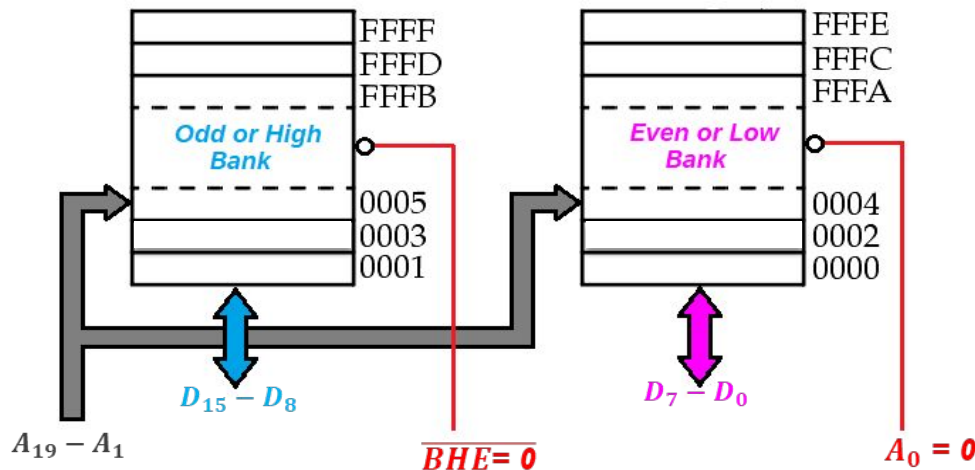
▸ Low bank disabled, High bank **enabled**



| | | Type of Transfer | Data Lines |
|---|---|---|---|
| 0 | 1 | Byte from odd bank | |

$A_{19} - A_1$  $\overline{BHE} = 0$  $A_0 = 1$

# 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 | |

Odd or High Bank — FFFF FFFD FFFB ... 0005 0003 0001

Even or Low Bank — FFFE FFFC FFFA ... 0004 0002 0000

$D_{15} - D_8$

$D_7 - D_0$

$A_{19} - A_1$

$\overline{BHE} = 0$

$A_0 = 0$

# 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 $D_7 - D_0$; $A_0 = 0$ & $\overline{BHE} = 1$
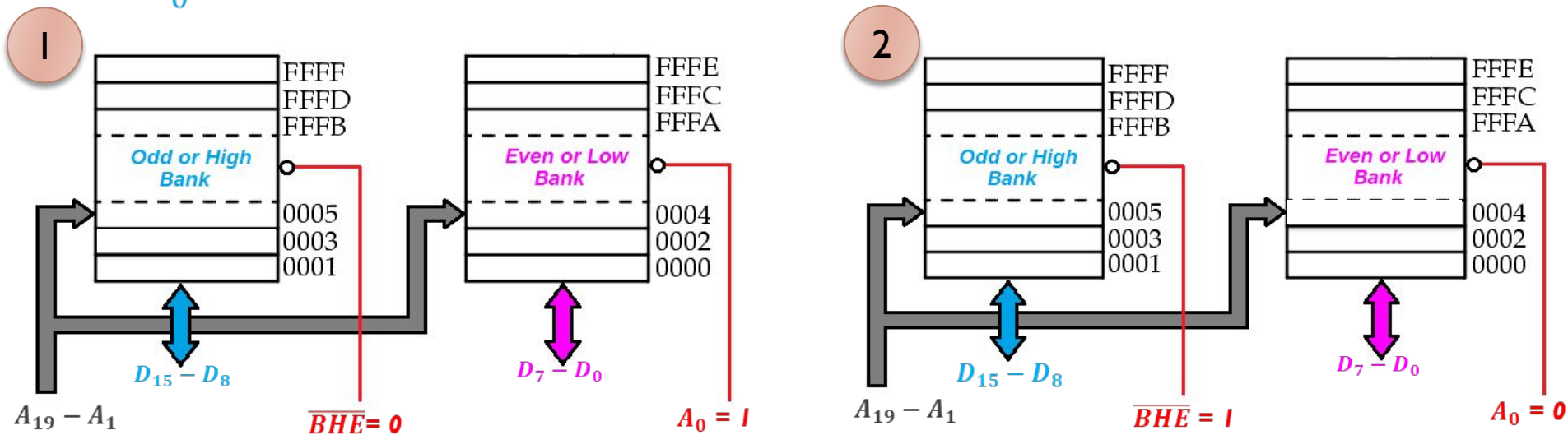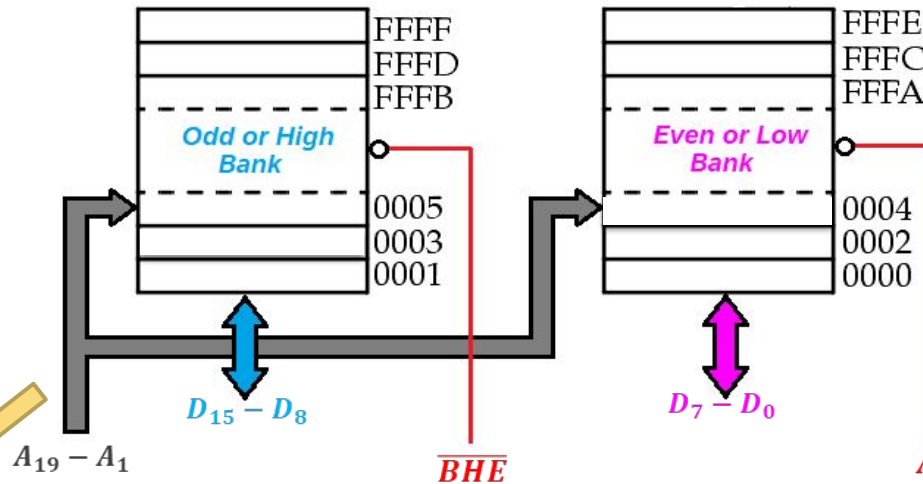
# An unaligned word
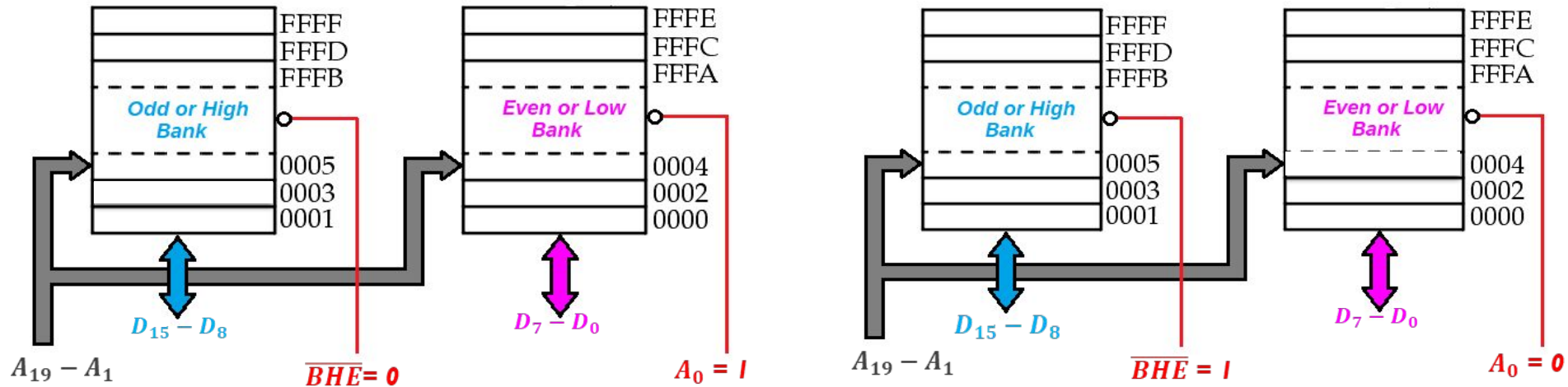*starting from an **odd** address*

**2 bus-cycles required**

**1st bus-cycle:**

odd addressed LSB via $D_{15} - D_8$

$A_0 = 1$ & $\overline{BHE} = 0$

FFFF
FFFD
FFFB

*Odd or High Bank*

0005
0003
0001

$D_{15} - D_8$

$A_{19} - A_1$

$\overline{BHE}$

FFFE
FFFC
FFFA

*Even or Low Bank*

0004
0002
0000

$D_7 - D_0$

$A_0$

**2nd bus-cycle:**

physical add auto-incremented

even addressed MSB via $D_7 - D_0$

$A_0 = 0$ & $\overline{BHE} = 1$

FFFF
FFFD
FFFB

*Odd or High Bank*

0005
0003
0001

$D_{15} - D_8$

$A_{19} - A_1$

$\overline{BHE} = 0$

FFFE
FFFC
FFFA

*Even or Low Bank*

0004
0002
0000

$D_7 - D_0$

$A_0 = 1$

FFFF
FFFD
FFFB

*Odd or High Bank*

0005
0003
0001

$D_{15} - D_8$

$A_{19} - A_1$

$\overline{BHE} = 1$

FFFE
FFFC
FFFA

*Even or Low Bank*

0004
0002
0000

$D_7 - D_0$

$A_0 = 0$

**CSE – 341: Microprocessors**
**BRAC University**

# Thank You