

# CG 2007 Microprocessor Systems

## Lecture 7

# Memory and I/O Interfacing

Dr. Ha Yajun  
(E1-08-13, *elehy@nus.edu.sg*)



# Lecture 7: Objectives and Outline

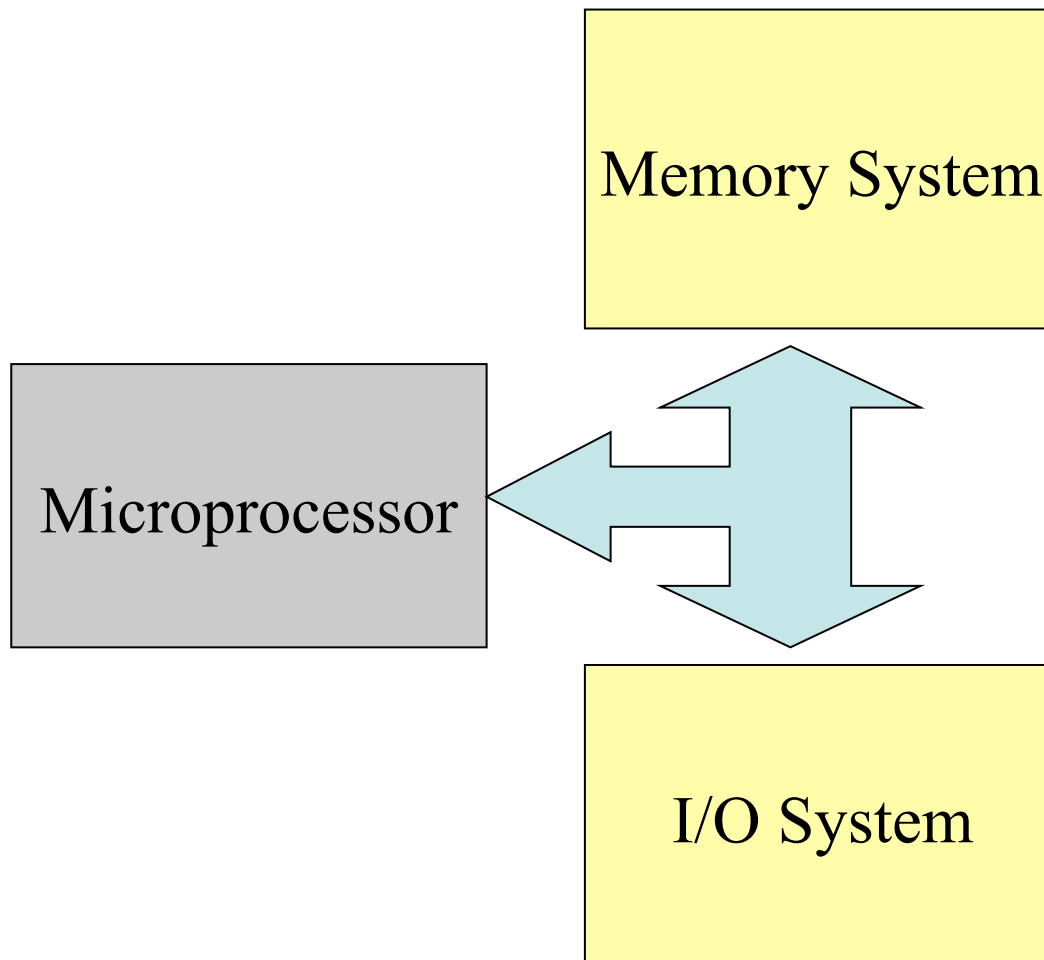
- **Objectives**

- Understand how to perform memory and I/O interfacing

- **Outline**

- **Module Overview**
- Memory and I/O Interfacing
- Interfacing Pins of the Intel 8088  $\mu$ P
- Memory and I/O Addressing
- 8088 vs 8086 Interfacing

# General Block Diagram of Microprocessor Systems



- **Memory system stores the instructions and data**
- **Microprocessor gets instructions and data from memory, executes instructions on data, and stores the processed data back to memory.**
- **I/O system extends the memory system to provide peripheral specific data to be processed by microprocessor.**
- **Our module studies three key words: Architecture, Programming and Interfacing**

# What is Inside a Real Microprocessor System?



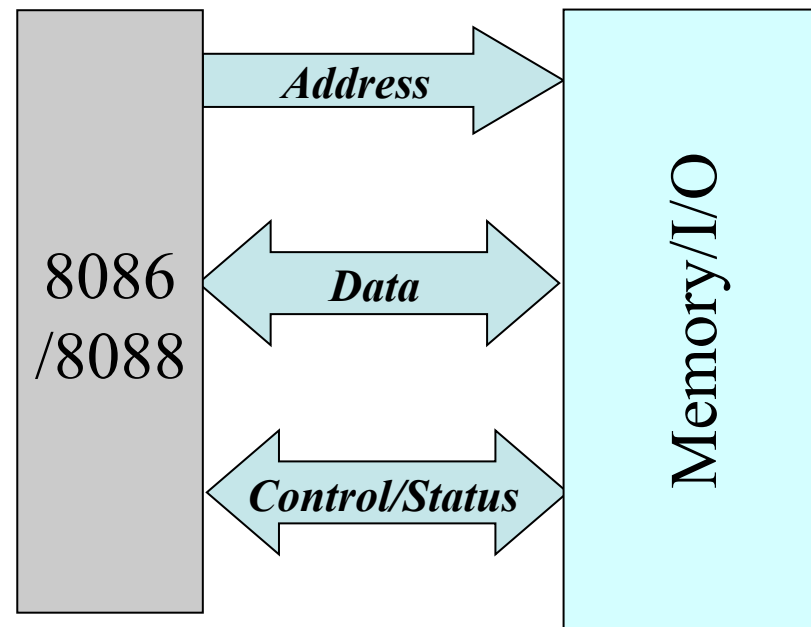
Link the real blocks in this [video](#) to the general block diagram in the previous slide!

The video can be accessed directly at <http://videos.howstuffworks.com/howstuffworks/23-computer-tour-video.htm>

# Lecture 7: Memory and I/O Interfacing

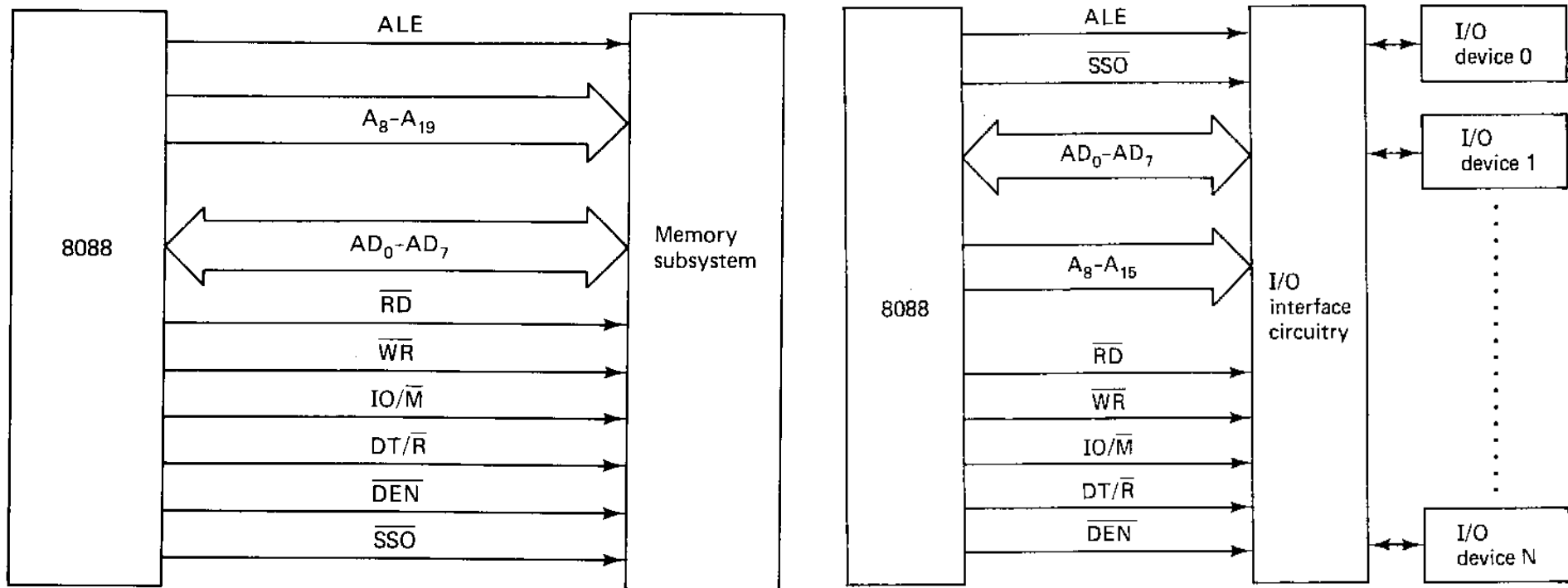
- Module Overview
- **Memory and I/O Interfacing**
- Interfacing Pins of the Intel 8088  $\mu$ P
- Memory and I/O Addressing
- 8088 vs 8086 Interfacing

# A First Look at Memory and I/O Interfacing



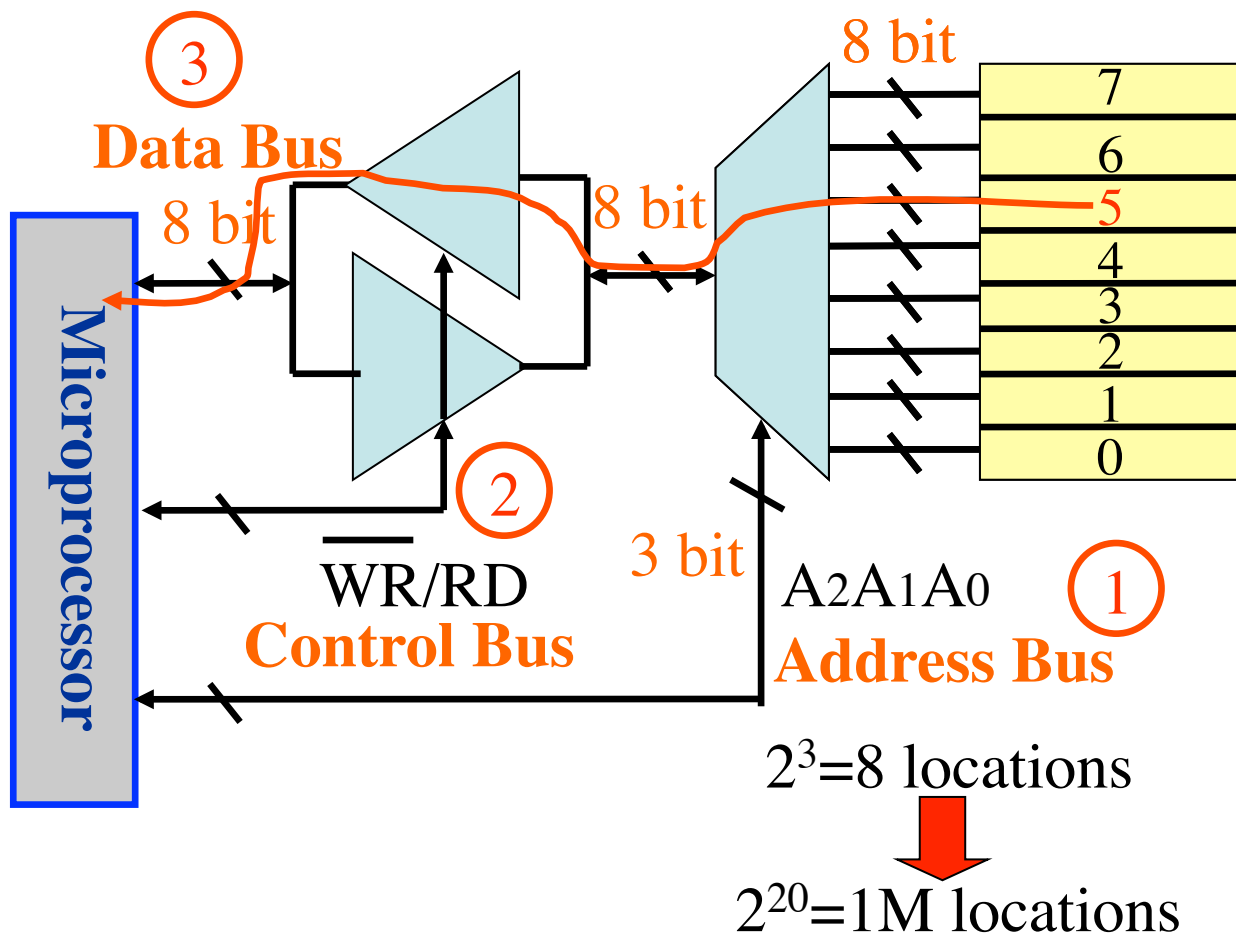
- $\mu$ P is interfaced to memory (I/O) to retrieve information and store results.
- Address bus decides which range of memory (I/O) space to be accessed by  $\mu$ P.
- Control bus decides READ or WRITE operation of the chosen memory (I/O) locations.
- Data Bus provides the data transfer channel between memory (I/O) and  $\mu$ P. For 8086, its data bus is 16 bit, but for 8088 its data bus is 8 bit.

# A Closer Look at Memory and I/O Interfacing



- Memory can be considered as a special I/O device or vice versa.
- Memory and I/O devices are interfacing with microprocessors with three types of signals:  
**Address, Data and Control.**

# A Small Memory Interfacing Example

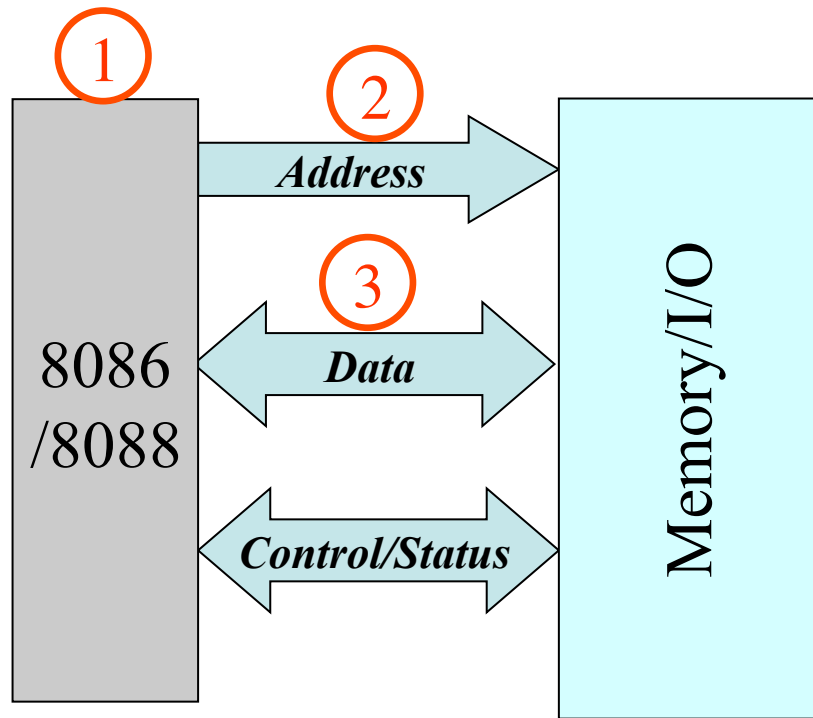


1. Address bus decides which location of memory to be accessed. The wider the address bus, the more locations can be addressed.
2. Control bus decides READ or WRITE operation of the chosen memory location.
3. Data Bus provides the data of the chosen locations. For 8086, its data bus is 16 bit, but for 8088 its data bus is 8 bit. The wider the data bus, the more locations can be accessed per operation.

**Quiz:** What are the widths of the data bus and address bus of your own computer?



# Issues to Consider During Interfacing



1. What are the **interfacing pins** that microprocessor is using.
2. How is a microprocessor **addressing** memory locations and I/O devices.
3. What's the differences between **8088 and 8086** interfacing.

**We discuss each of the three issues in the following three sections respectively.**

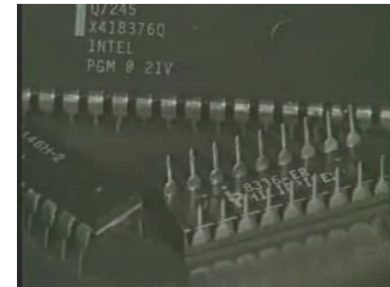
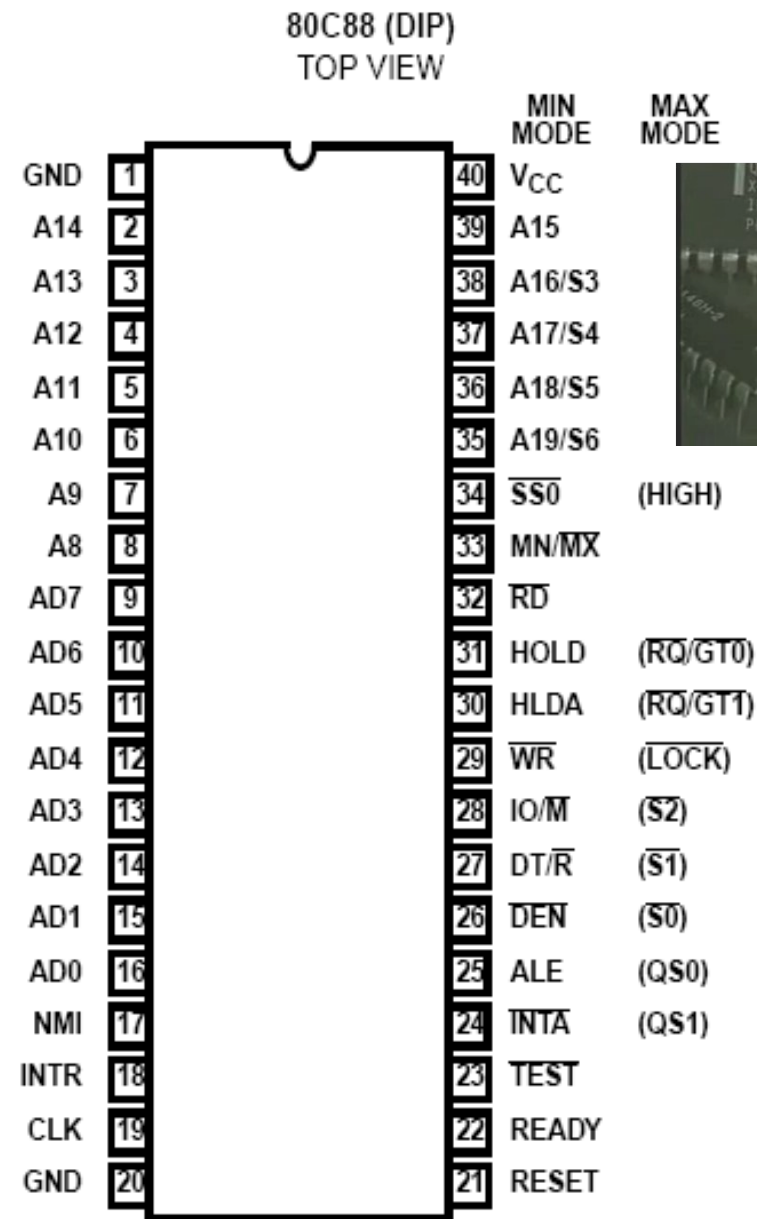
# Lecture 7: Memory and I/O Interfacing

- Module Overview
- Memory and I/O Interfacing
- Interfacing Pins of the Intel 8088  $\mu$ P
- Memory and I/O Addressing
- 8088 vs 8086 Interfacing

# Interfacing Pins of an Intel 8088 Microprocessor

- **40-pin chip**
  - Data/address Pins
  - Control Pins
  - Status Pins
  - Interrupt Pins
  - DMA Interface Pins
  - Mode Pins

All the above pins are not required to memorize! You just need to know their functions.



[Video](#) on microprocessor packaging and testing

# Interfacing Pins of an Intel 8088 Microprocessor

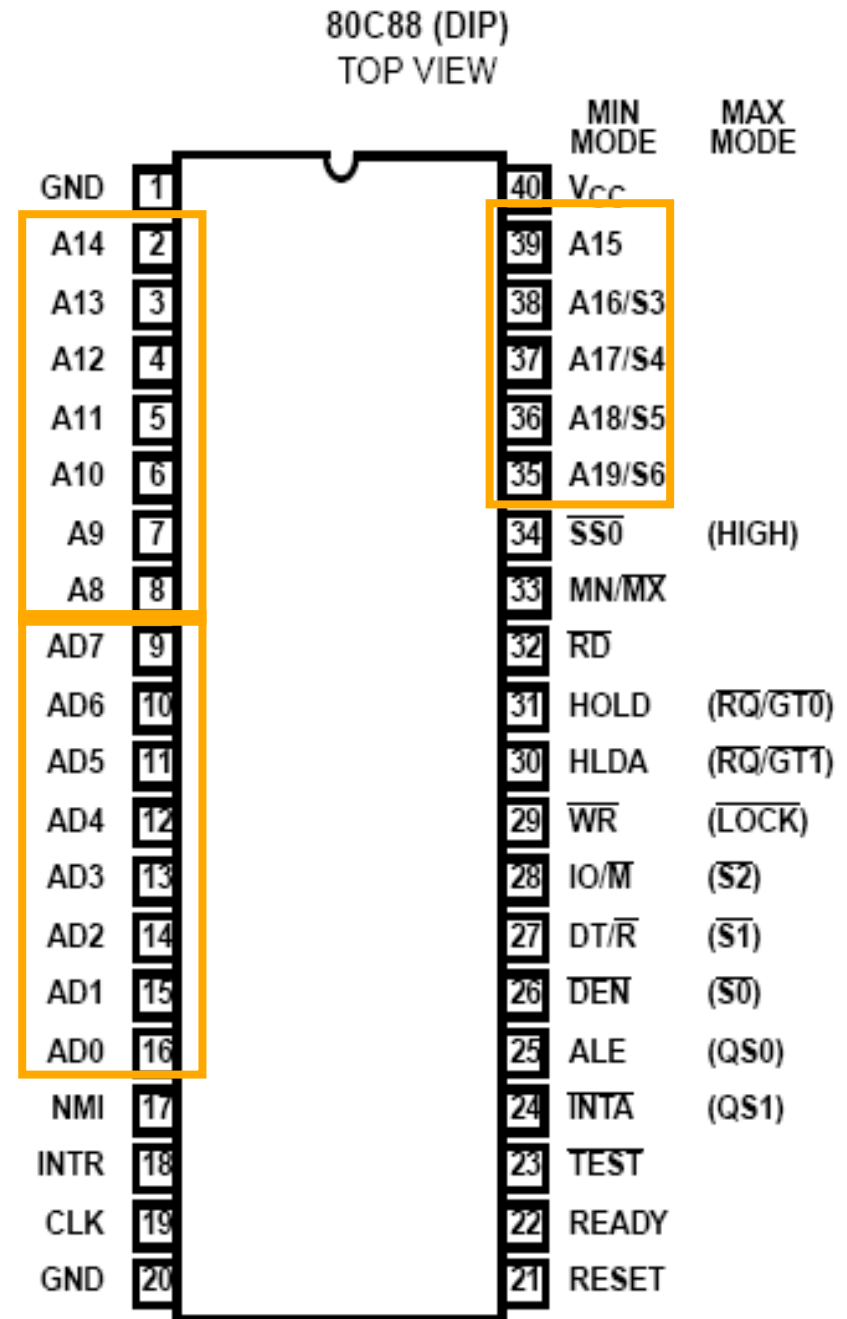
- **Data Bus Pins:**

- $AD_7..AD_0$ : multiplexed with address lines  $A_7..A_0$

- **Address Bus Pins:**

- $AD_7..AD_0$  multiplexed with data lines  $D_7..D_0$
  - $A_8..A_{15}$
  - $A_{19}/S_6..A_{16}/S_3$ : multiplexed with status signals  $S_6..S_3$

Quiz: Why are  $A_7..A_0$  and  $D_7..D_0$  shared?



# Interfacing Pins of an Intel 8088 Microprocessor

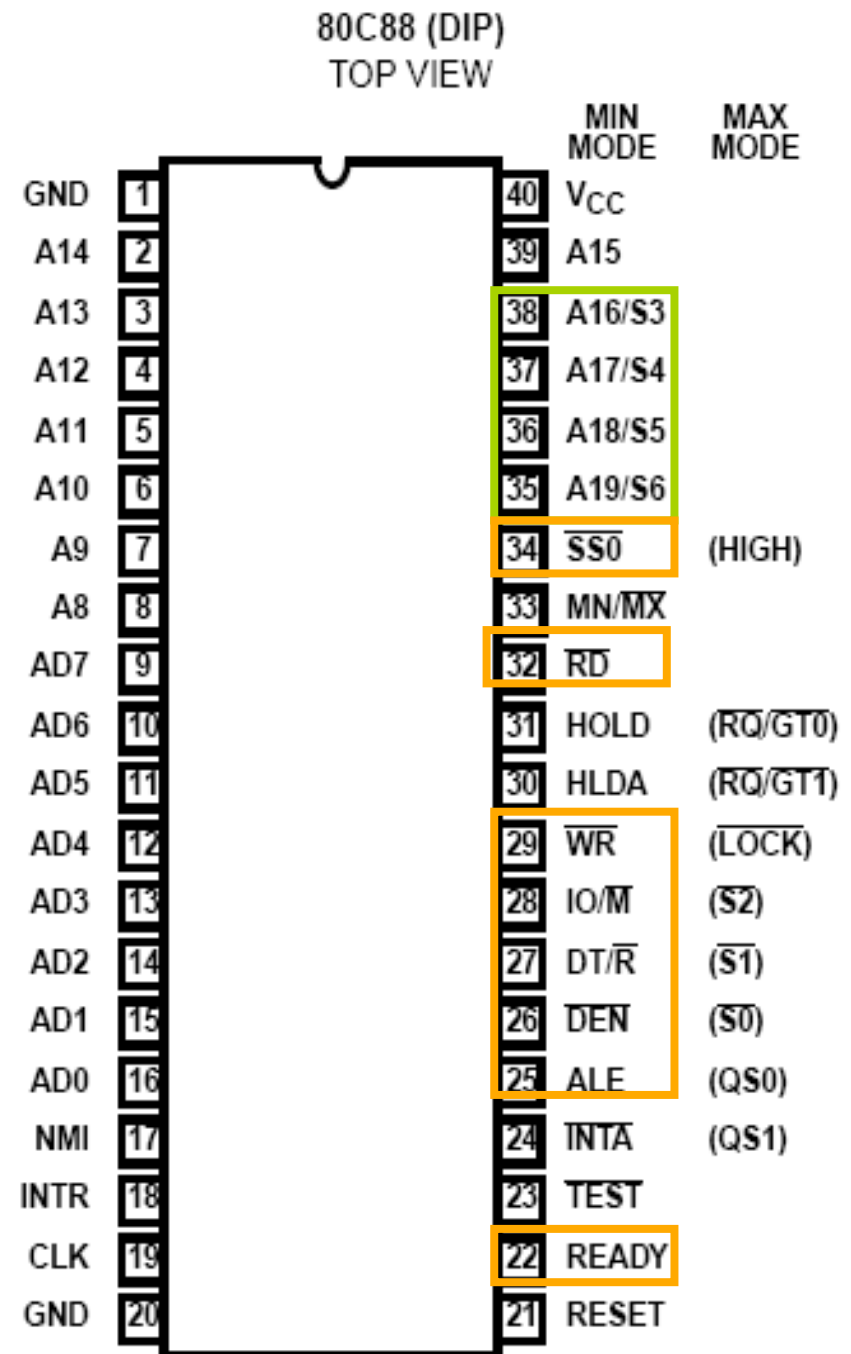
## • Status Signal Pins:

- $A_{19}/S_6 \dots A_{16}/S_3$ :  
 $S_4 \dots S_3$ : identifies segments,  
 $S_5$ : IF flag,  
 $S_6$ : not used, tied low

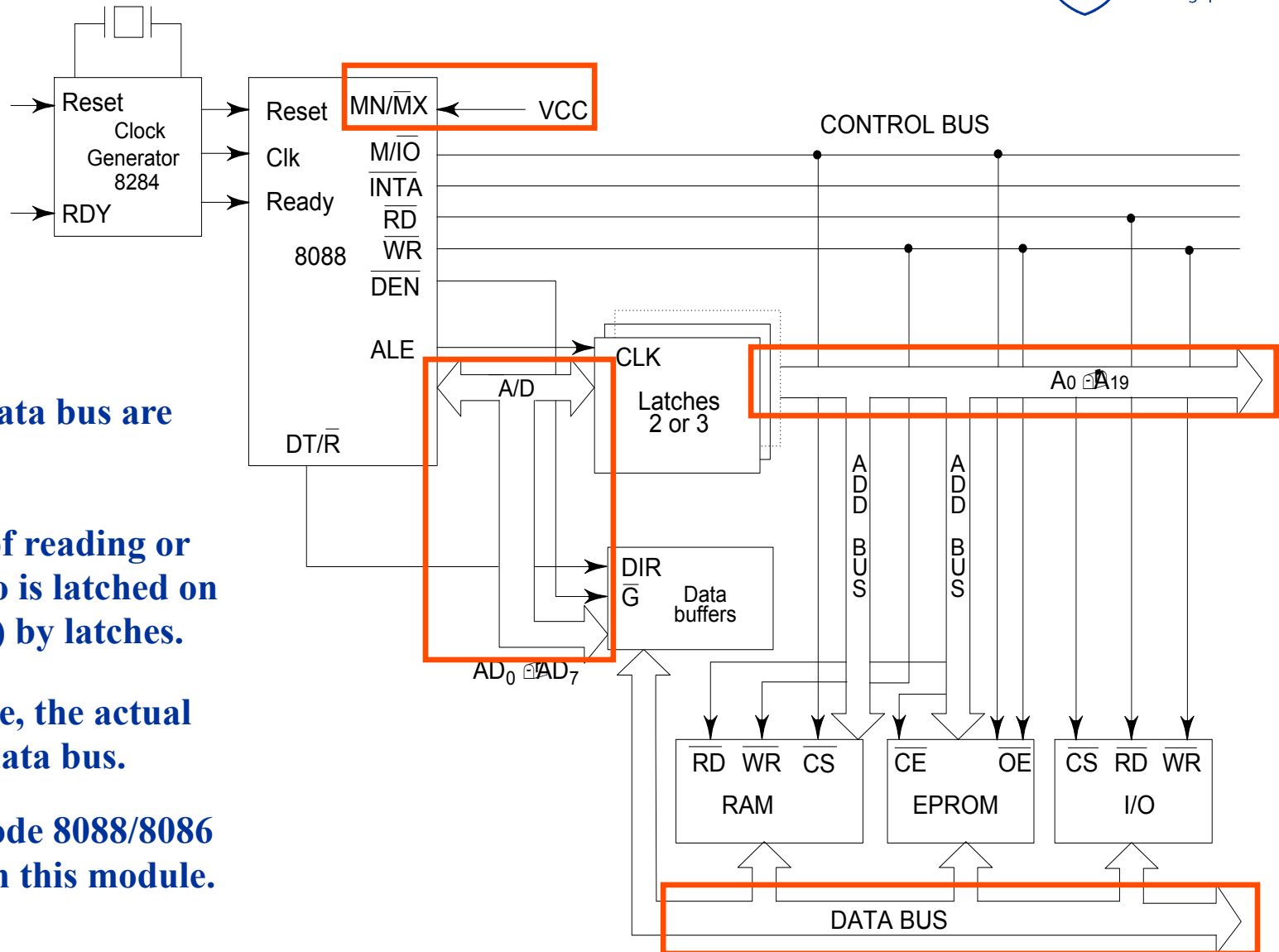
## • Control Bus Pins:

- ALE: address latch enable
- $IO/\overline{M}^*$ ,  $DT/\overline{R}^*$ ,  $\overline{SSO}$
- $\overline{RD}^*$ ,  $\overline{WR}^*$ ,  $\overline{DEN}^*$  (data enable),  $\overline{READY}$

**X\*: active low signal**



# Minimum Mode Intel 8088 Microprocessor System



- Address bus and data bus are time multiplexed.
- In the first phase of reading or writing, address info is latched on address bus (A<sub>0</sub>-A<sub>19</sub>) by latches.
- In the second phase, the actual data info is sent to data bus.
- Only minimum mode 8088/8086 system is required in this module.

# Interfacing Pins of an Intel 8088 Microprocessor

## • Interrupt controller Pins:

- INTR, INTA\*
- TEST\*
- NMI, RESET

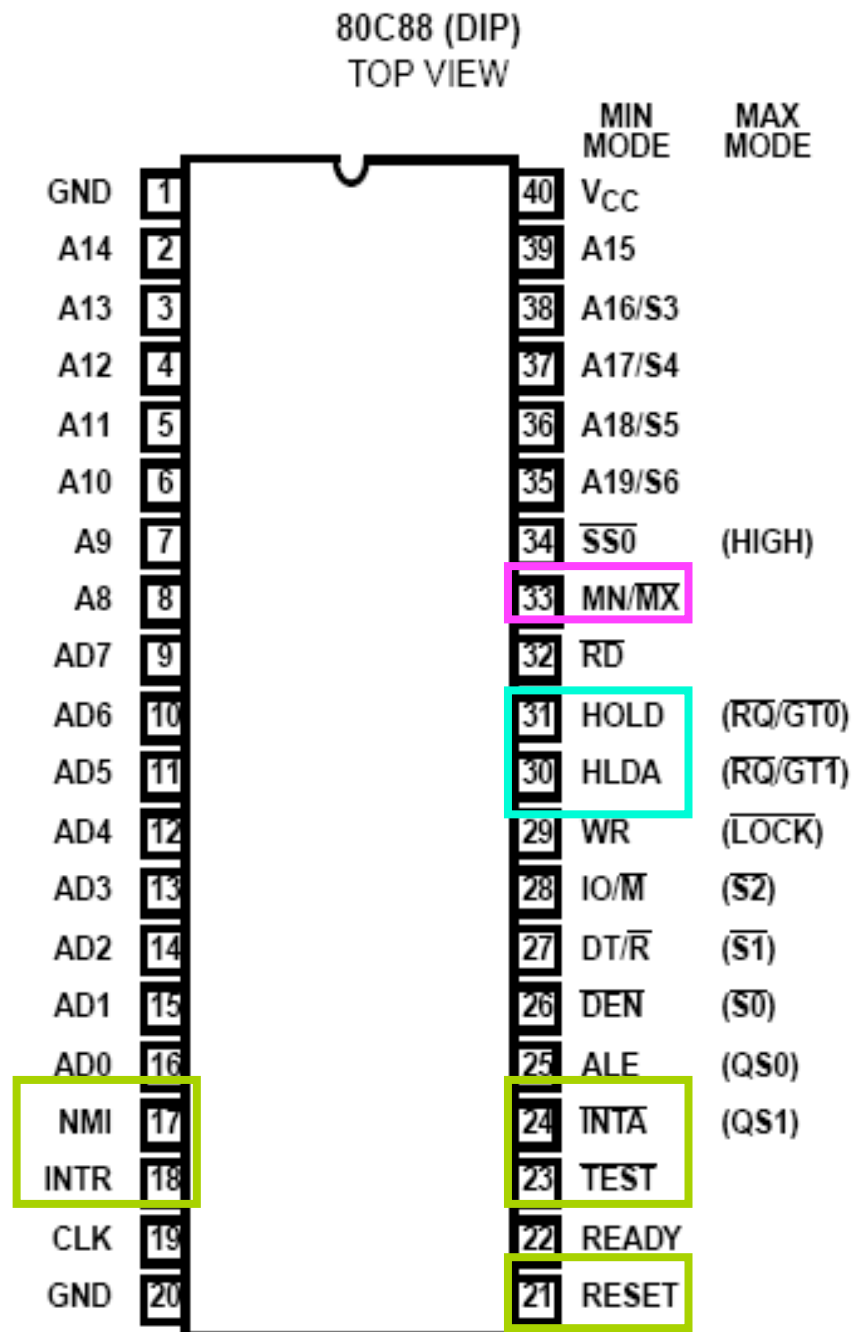
## • DMA interface Pins

- HOLD, HLDA

## • Mode Pin

- MN/MX\*: single vs multiple processors

Interrupt and DMA pins will be introduced in detail in lectures 9 & 10.



# Lecture 7: Memory and I/O Interfacing

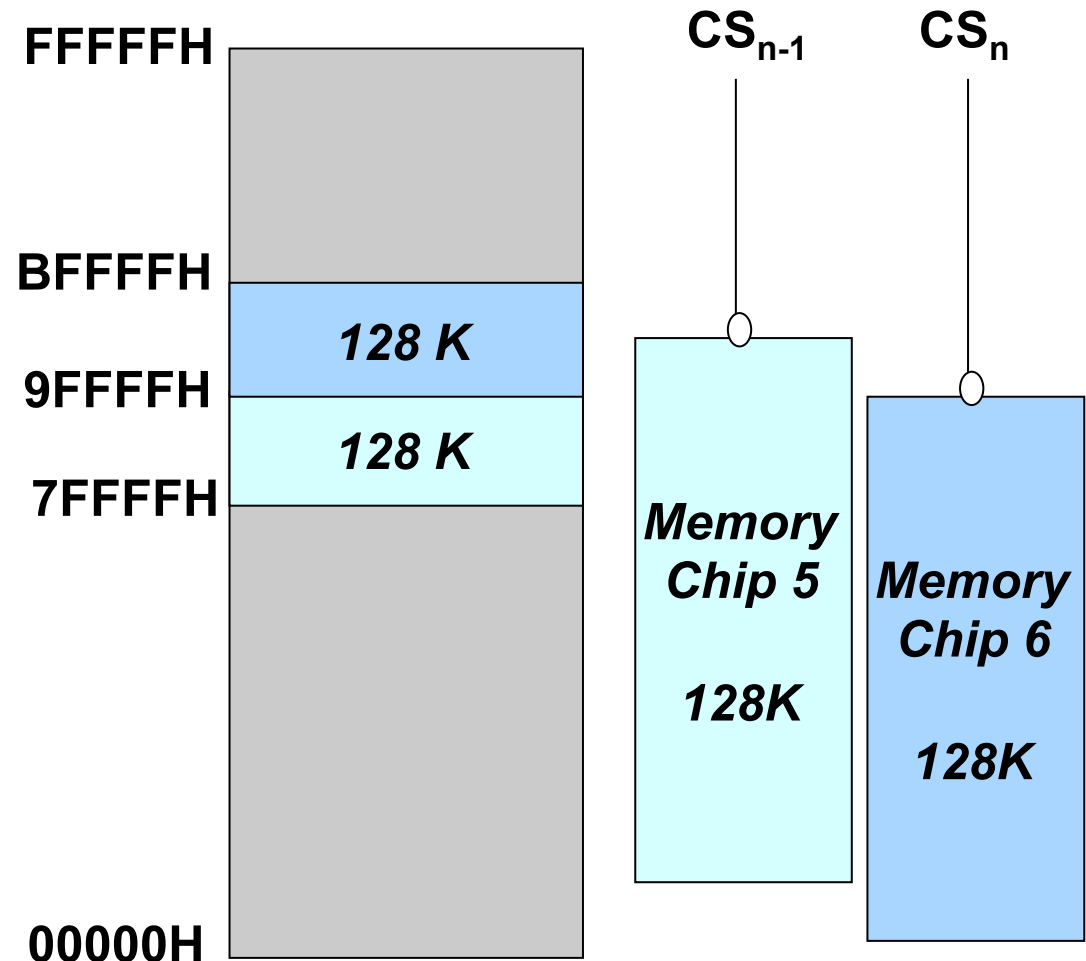
- Module Overview
- Memory and I/O Interfacing
- Interfacing Pins of the Intel 8088  $\mu$ P
- **Memory and I/O Addressing**
- 8088 vs 8086 Interfacing



# Memory Address Space vs Actual Memory Size

The memory address space is  $2^n$ , where  $n$  is the number of address bus lines. In 8088/8086,  $n=20$ , thus its memory address space is 00000H-FFFFFFH.

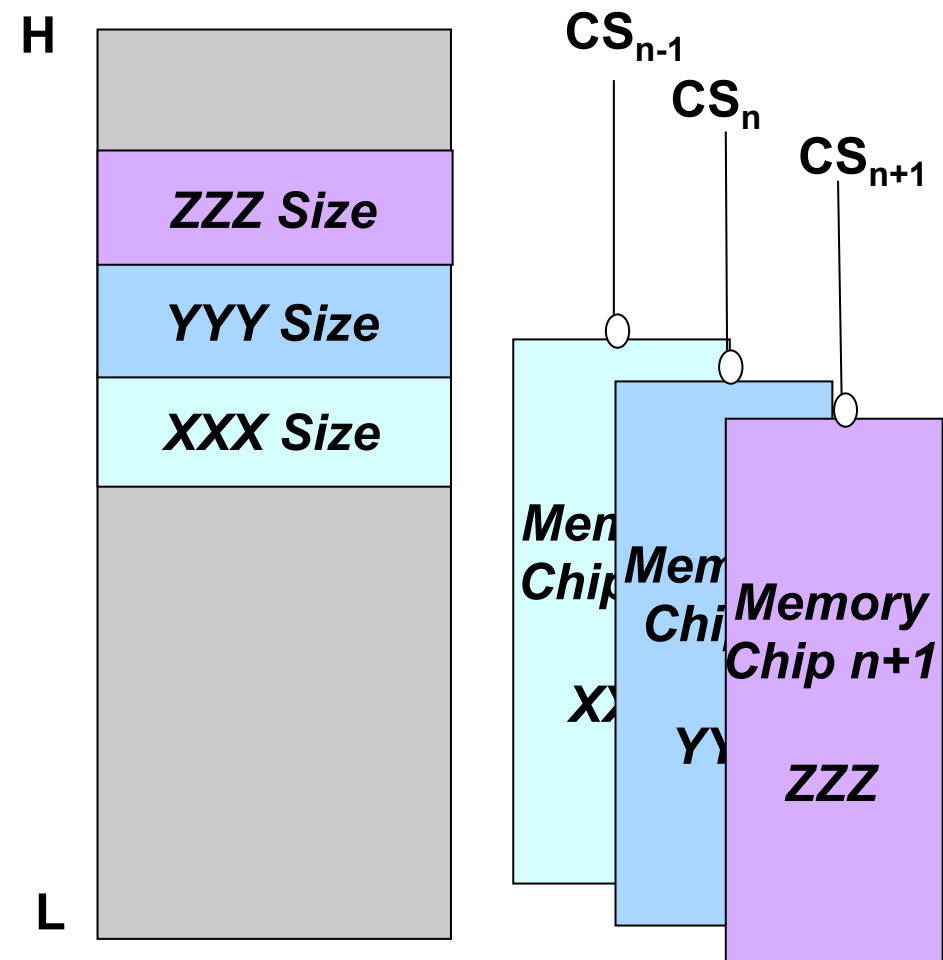
The actual memory size depends on how many memory locations have been covered by memory chips in the memory address space. In this example, the actual memory size is 256K.



# How to Increase Memory Size of Your Computer

Although the memory address space of a computer has been fixed by its address bus size, buying and installing as many as possible more memory chips will increase the actual memory size of your computer.

Please see the [video](#) to see how you can increase the actual memory size of your computer.



Video on <http://videos.howstuffworks.com/howstuffworks/26-how-to-install-ram-to-a-desktop-computer-video.htm>

# Memory Addressing Issues

1. Decide the **memory address map**. It shows which type of memory chip is mapped to which address range.

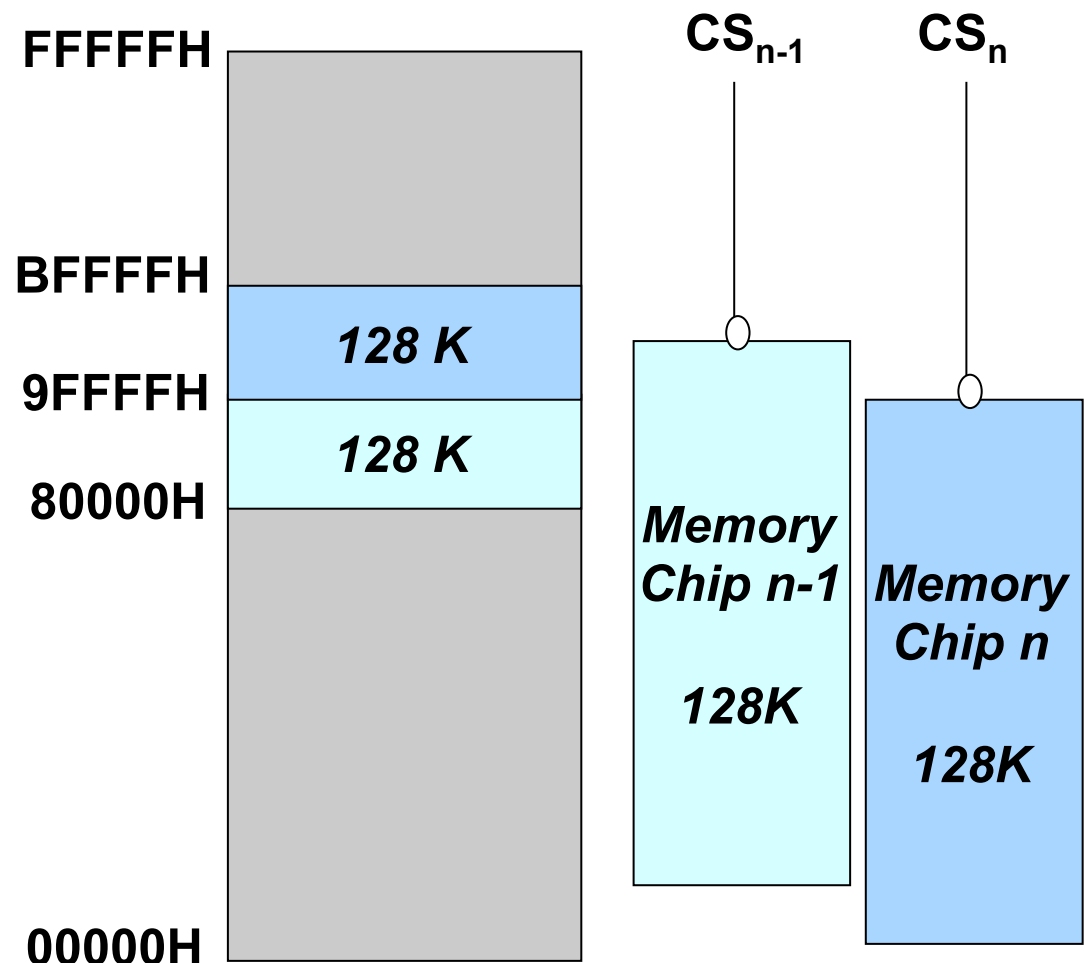
Chip N-1: 80000H-9FFFFH

Chip N: A0000H-BFFFFH

2. Derive the **logic function of chip select signal  $CS_n$**  for each memory chip based on the memory address map.

$$CS_{n-1} = A_{19}A_{18}\overline{A_{17}}$$

$$CS_n = A_{19}A_{18}\overline{A_{17}}$$



# Lecture 7: Memory and I/O Interfacing

- Module Overview
- Memory and I/O Interfacing
- Interfacing Pins of the Intel 8088  $\mu$ P
- Memory and I/O Addressing
  - **Decide the Memory Address Map**
  - **Derive the Chip Select Logic Function**
- 8088 vs 8086 Interfacing

# Memory Address Mapping

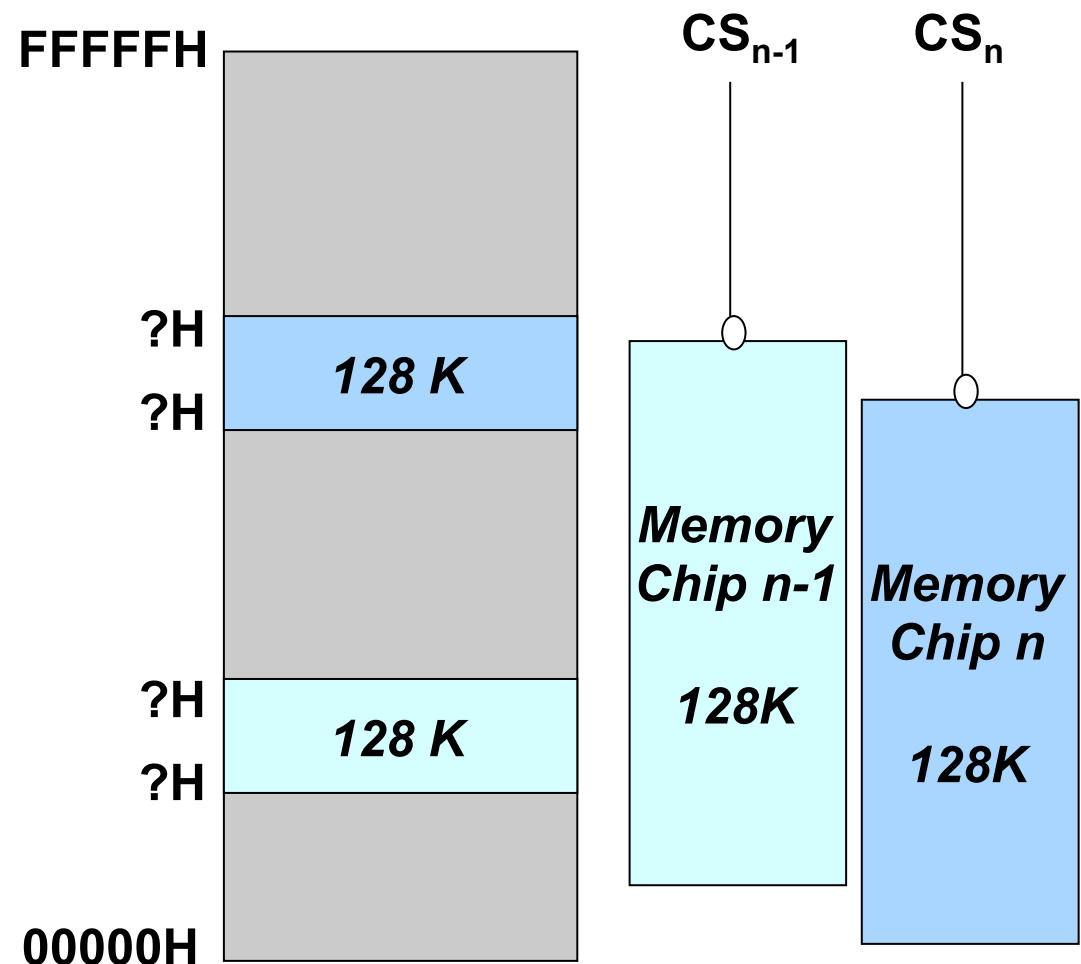
**Memory address mapping**  
decides:

1. which type of memory chip (RAM or ROM)
2. should be mapped to which address range?

Address Mapping Results:

Chip N-1: Type, Range

Chip N: Type, Range



# ROM vs RAM



- **ROM** stands for read-only memory. A ROM chip will maintain its information even when power goes off.

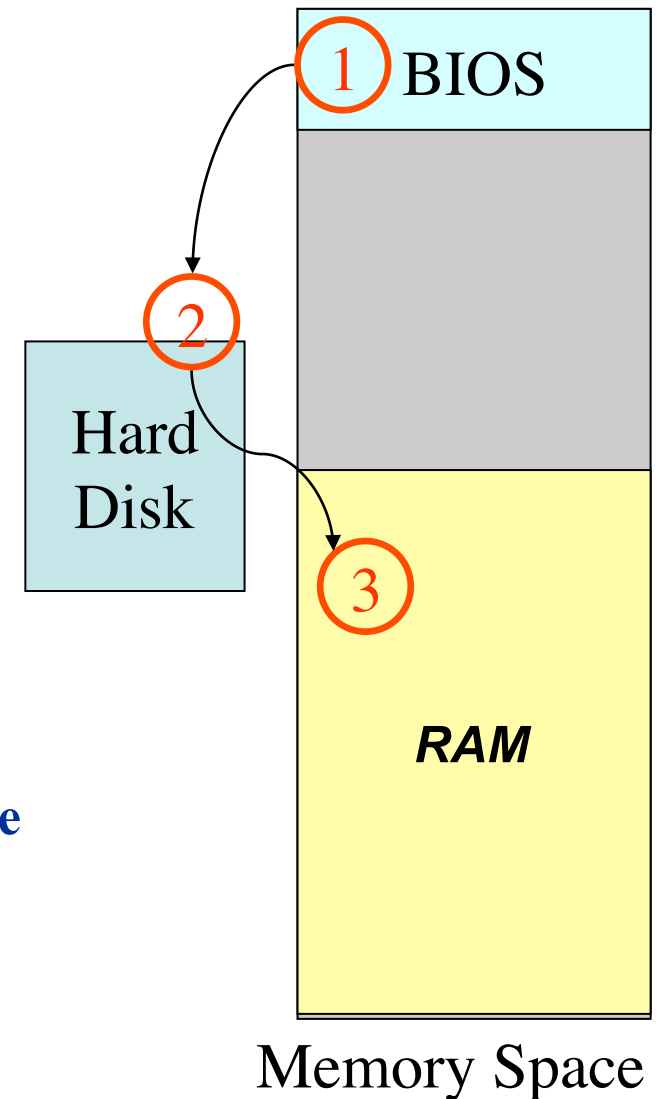


- **RAM** stands for random-access memory. RAM contains bytes of information, and the microprocessor can read or write to those bytes depending on whether the RD or WR line is signaled. One problem with today's RAM chips is that they forget everything once the power goes off. That is why the computer needs ROM.

# How a Computer Loads Data When It Starts

Nearly all computers contain some amount of ROM and RAM. On a PC, the ROM is called the **BIOS** (Basic Input/Output System).

1. When the microprocessor starts, it begins executing instructions it finds in the BIOS. The BIOS instructions do things such as testing the hardware in the machine.
2. Then it goes to the hard disk to fetch the boot sector. This boot sector is another small program, and the BIOS stores it in RAM after reading it off the disk.
3. The microprocessor then begins executing the boot sector's instructions from RAM. The boot sector program will tell the microprocessor to fetch something else from the hard disk into RAM, which the microprocessor then executes, and so on.



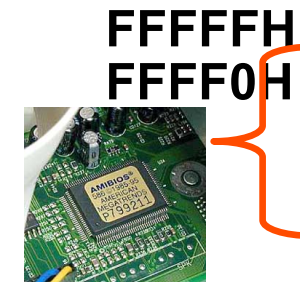
# Two Constraints in Memory Address Map

- **Where is the reset vector of the microprocessors**

- in case of the Intel 8086: CS:IP = FFFF0h

**Reset vector stores the first instruction to be executed in BIOS.**

- Requires at least one EPROM/ROM chip to be mapped at the highest addresses



**Reset Vector**

- **Where is the interrupt vector table stored?**

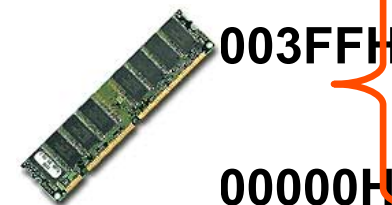
- in case of the Intel 8086: at 00000H-003FFh

**The first 1K locations of memory address space**

- Requires at least one RAM chip to be mapped at the lowest addresses

- **For the simplest system, requires**

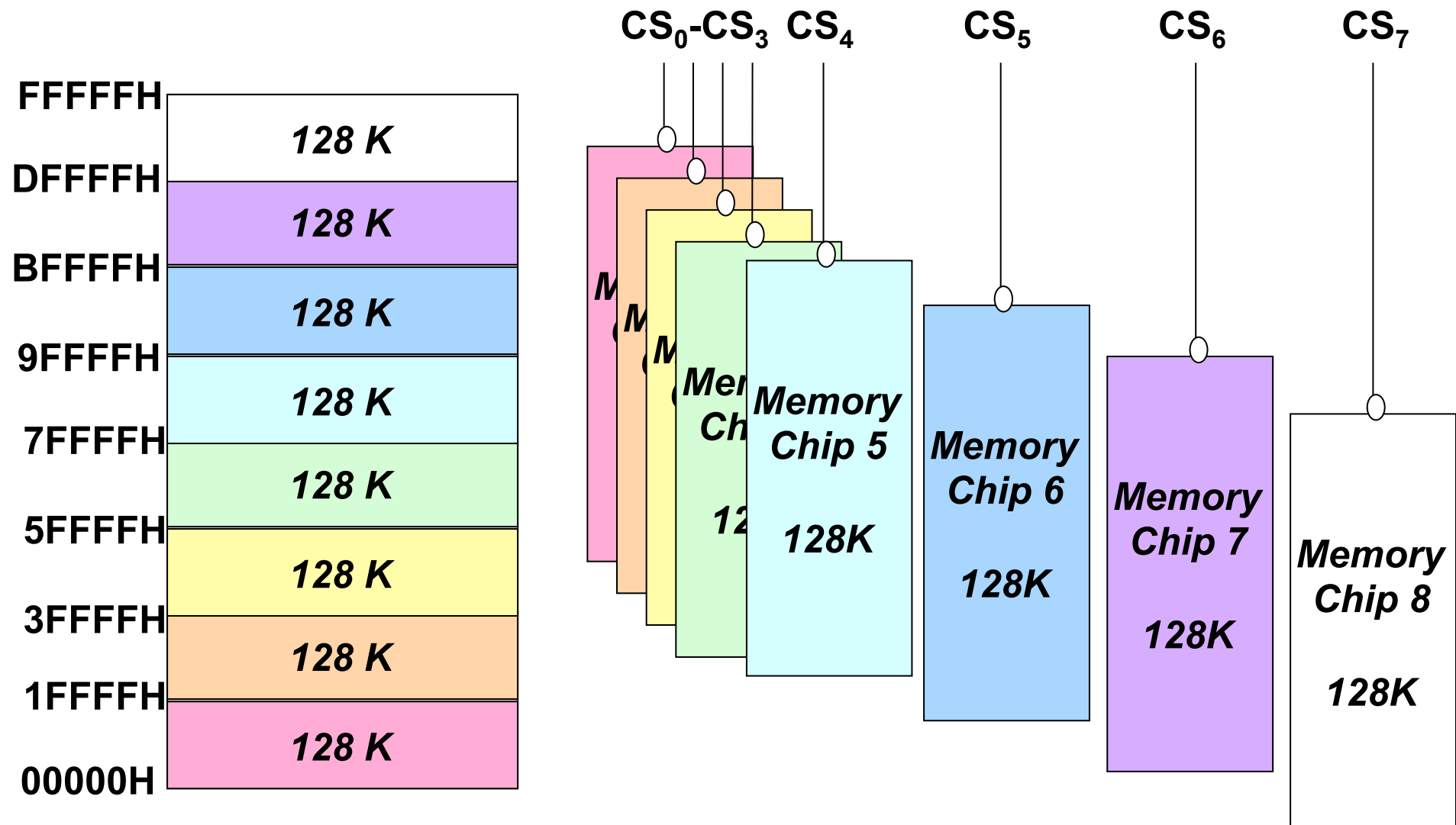
- 1 EPROM chip from FFFFFh downward
  - 1 RAM chip from 00000h upward



**Interrupt Vector Table**

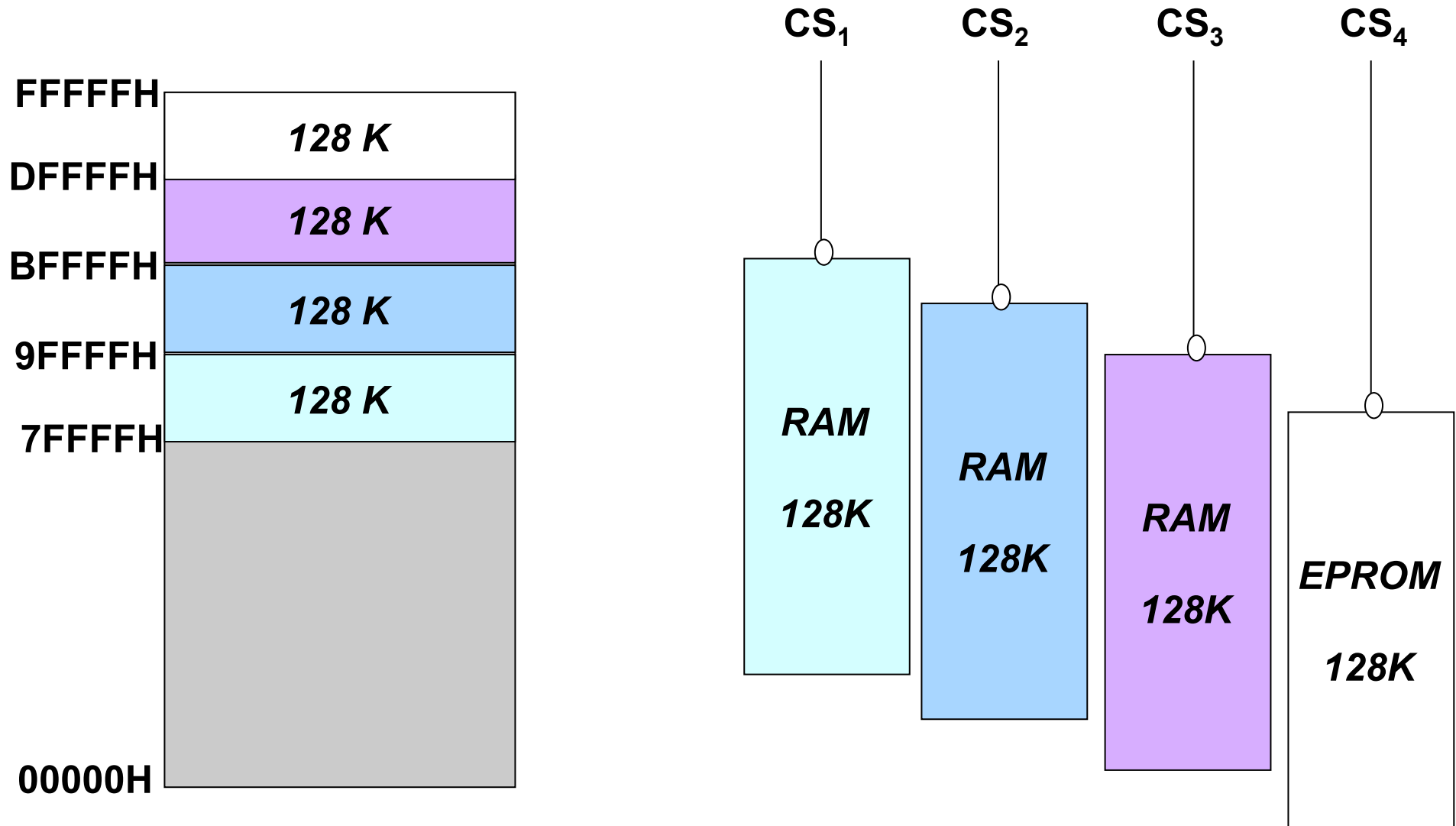


# Memory Address Map Quiz 1



**Quiz: Which of the 8 memory chips must be a ROM/EPROM?**

# Memory Address Map Quiz 2



Quiz: Is this a working memory address map?

# Lecture 7: Memory and I/O Interfacing

- Module Overview
- Memory and I/O Interfacing
- Interfacing Pins of the Intel 8088  $\mu$ P
- Memory and I/O Addressing
  - **Decide the Memory Address Map**
  - **Derive the Chip Select Logic Function**
- 8088 vs 8086 Interfacing

# Derive the Chip Select Signal

1. Suppose the following **memory address map**:

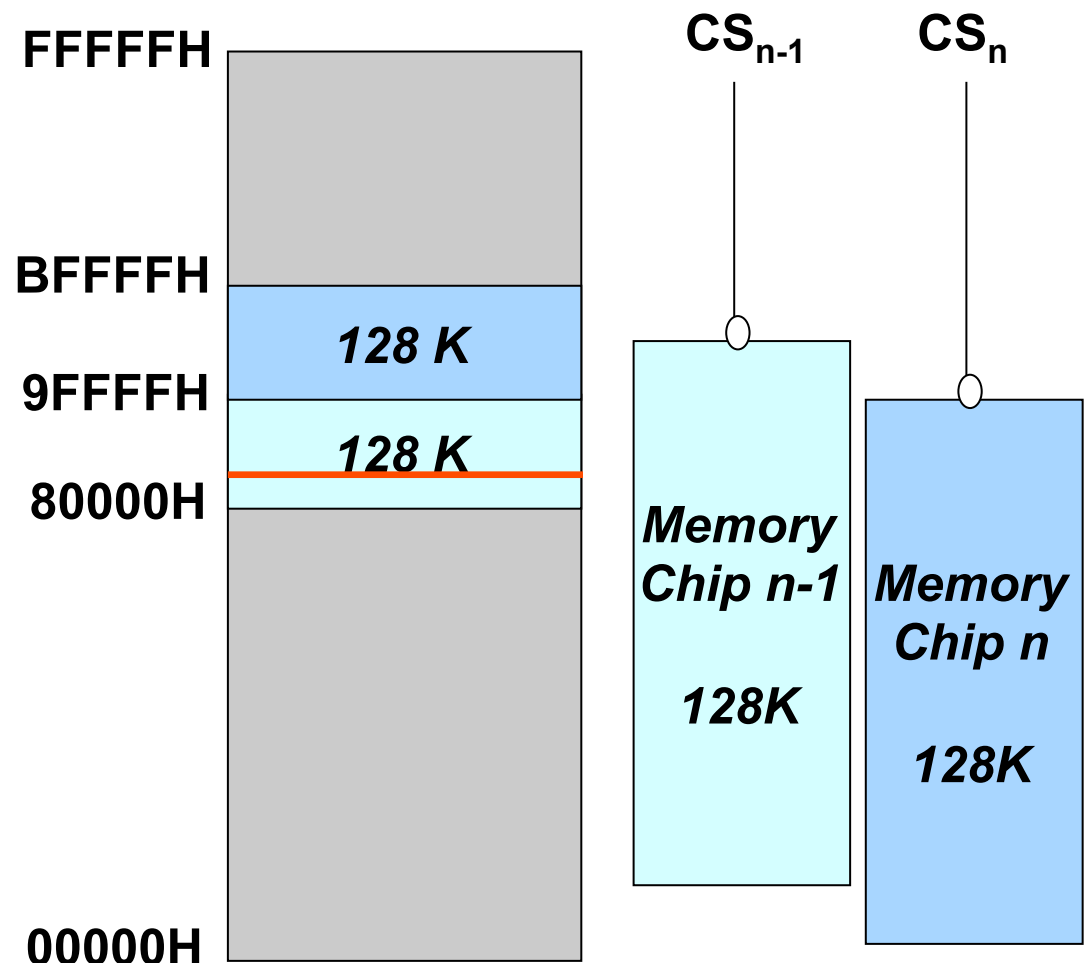
Chip N-1: 80000H-9FFFFH

Chip N: A0000H-BFFFFH

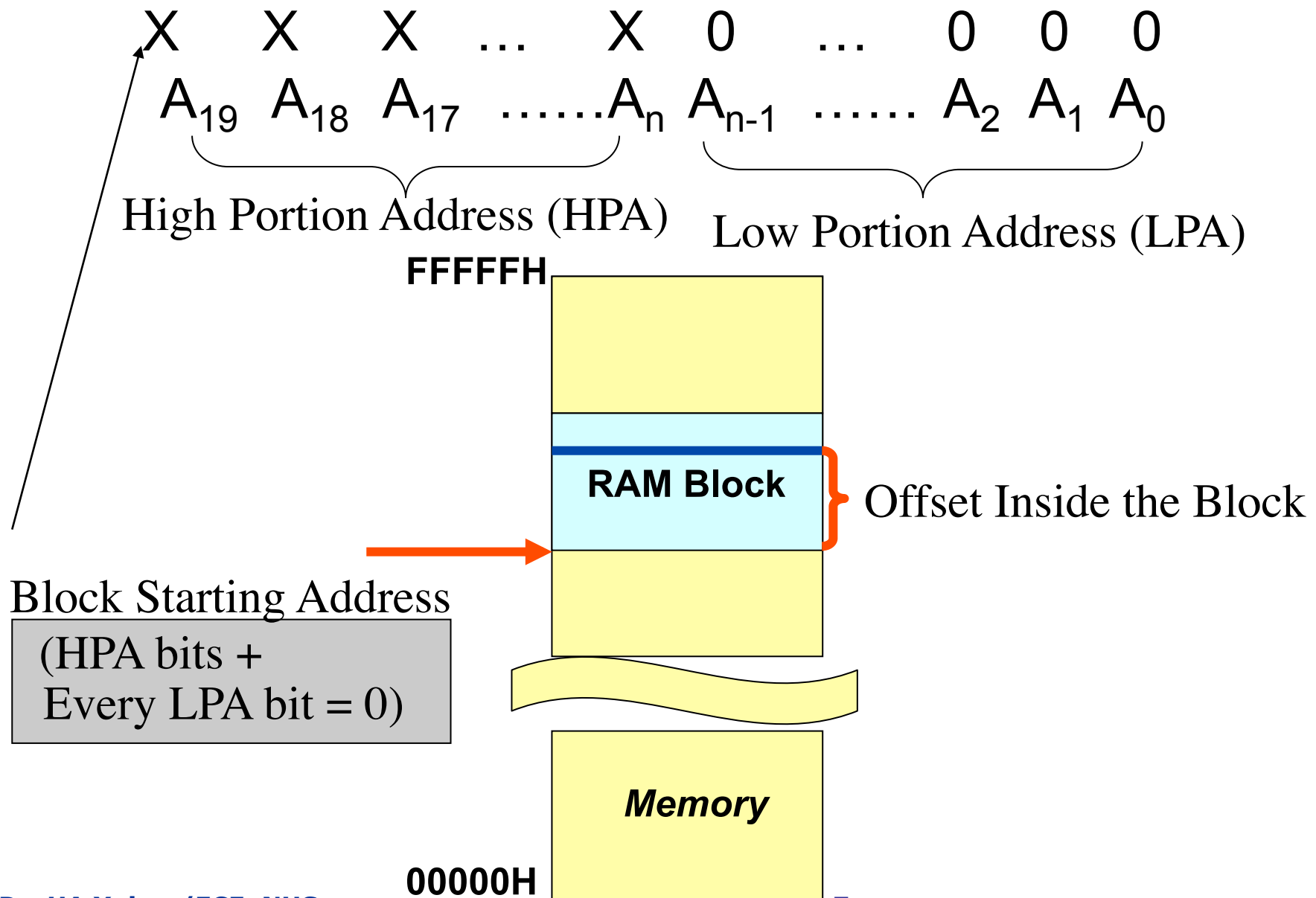
If you want to read the data at 80028H, which memory chip should be activated?

2. The **logic function of chip select signal  $CS_n$**  for each memory chip answers this type of question.

$$CS_{n-1} = A_{19} \wedge A_{18} \wedge A_{17}$$

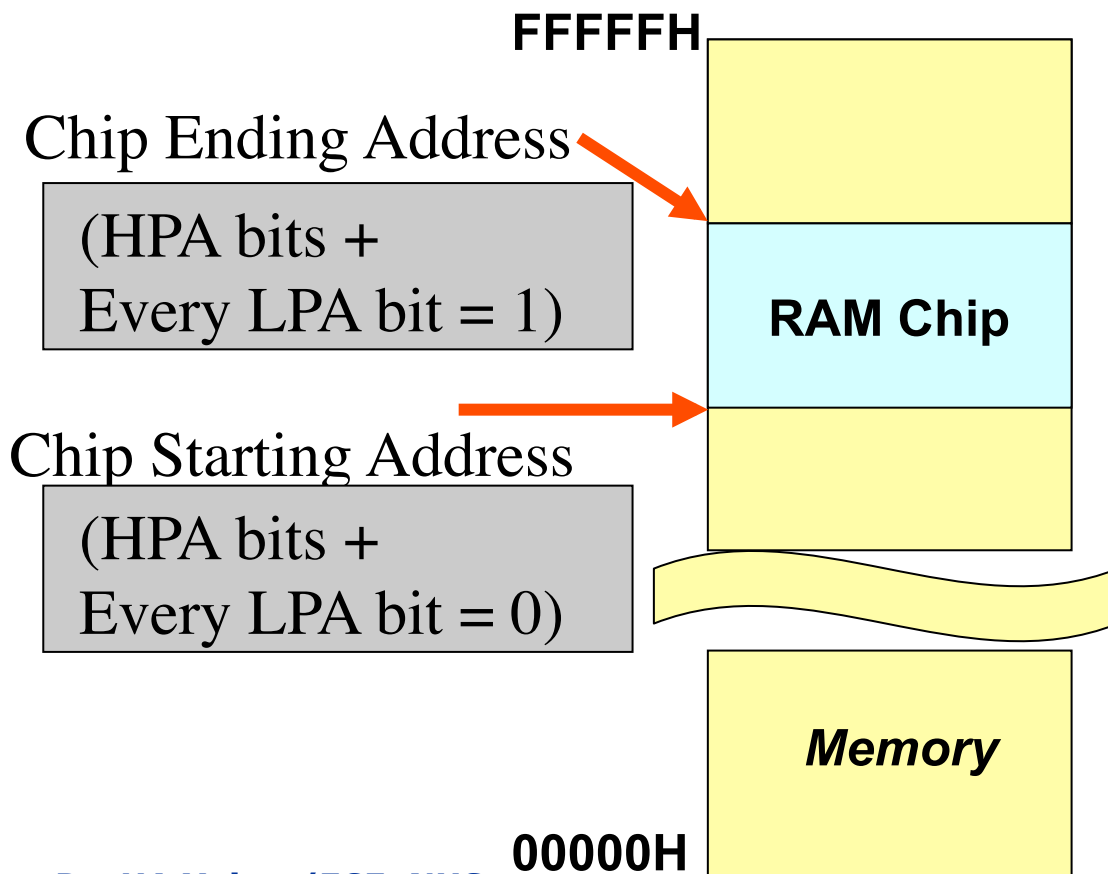


# A Two-Portion Memory Address



# Chip Select Signal Logic Function

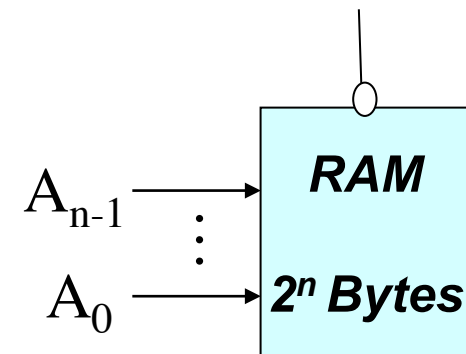
$A_{19} \ A_{18} \ A_{17} \ \dots \ A_n \ A_{n-1} \ \dots \ A_2 \ A_1 \ A_0$   
 High Portion Address (HPA) Low Portion Address (LPA)



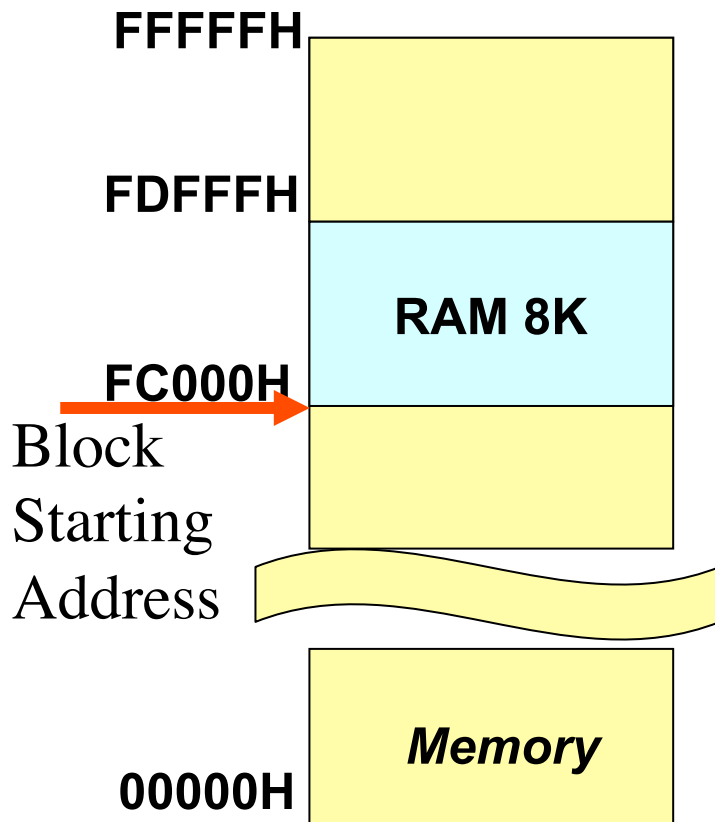
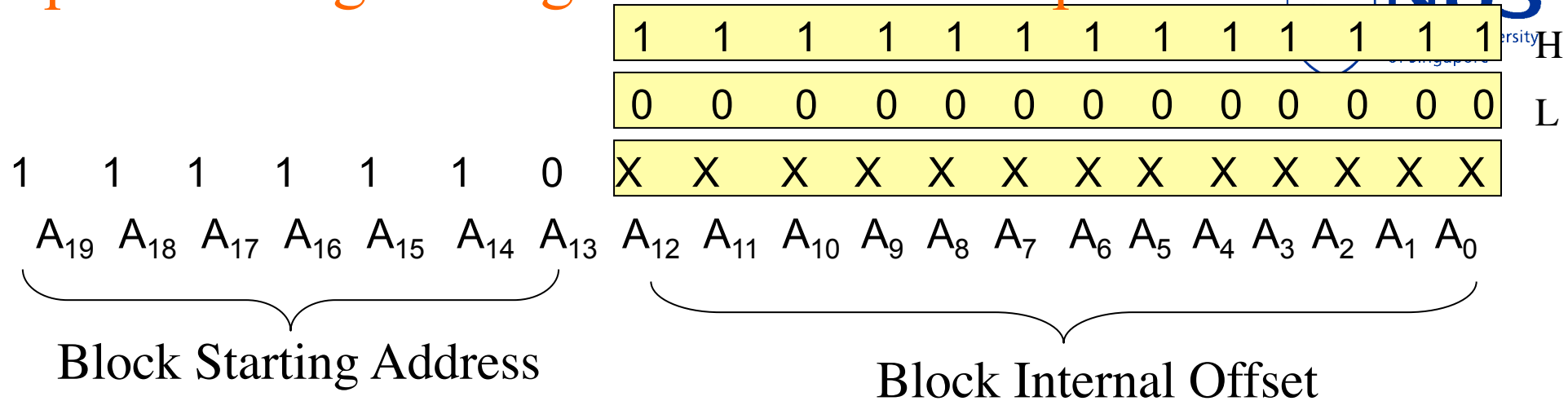
When an RAM chip is mapped to an address range, its CS\ logic function is:

$$\overline{\text{CS}} = f(\text{HPA})$$

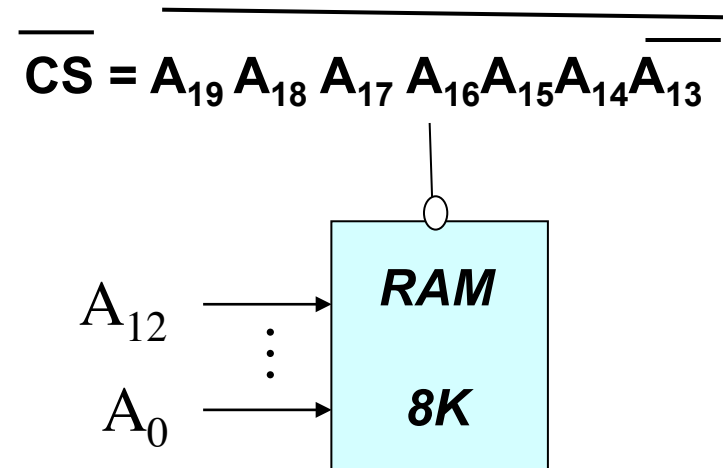
$$\overline{\text{CS}} = f(A_{19}-A_n)$$



# Chip Select Signal Logic Function Example



When the RAM chip is mapped to the address range FC000H-FDFFFH, its CS\ logic function is:



# Lecture 7: Memory and I/O Interfacing

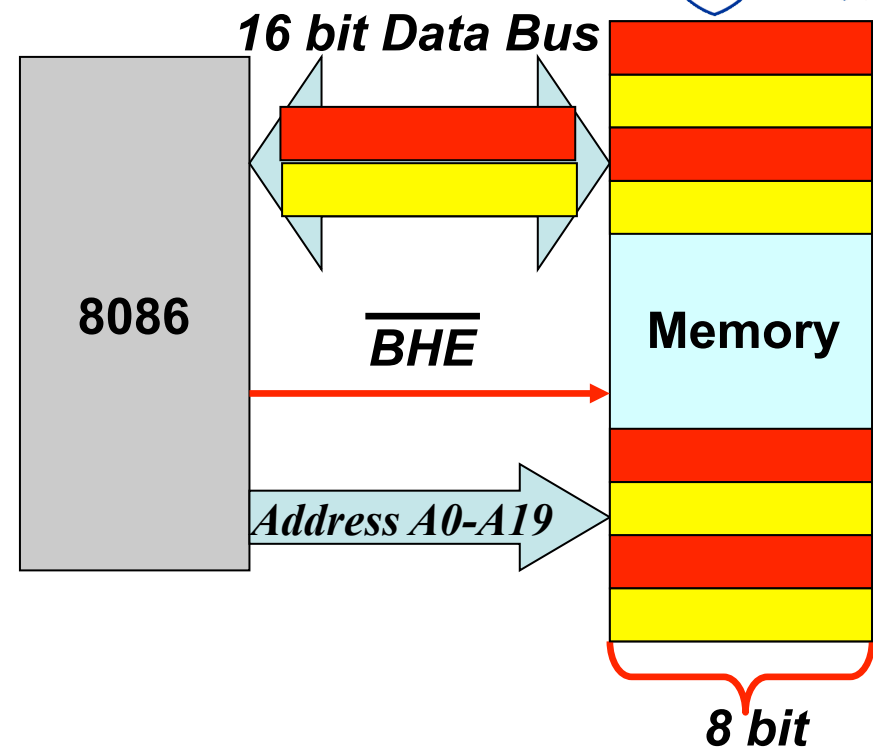
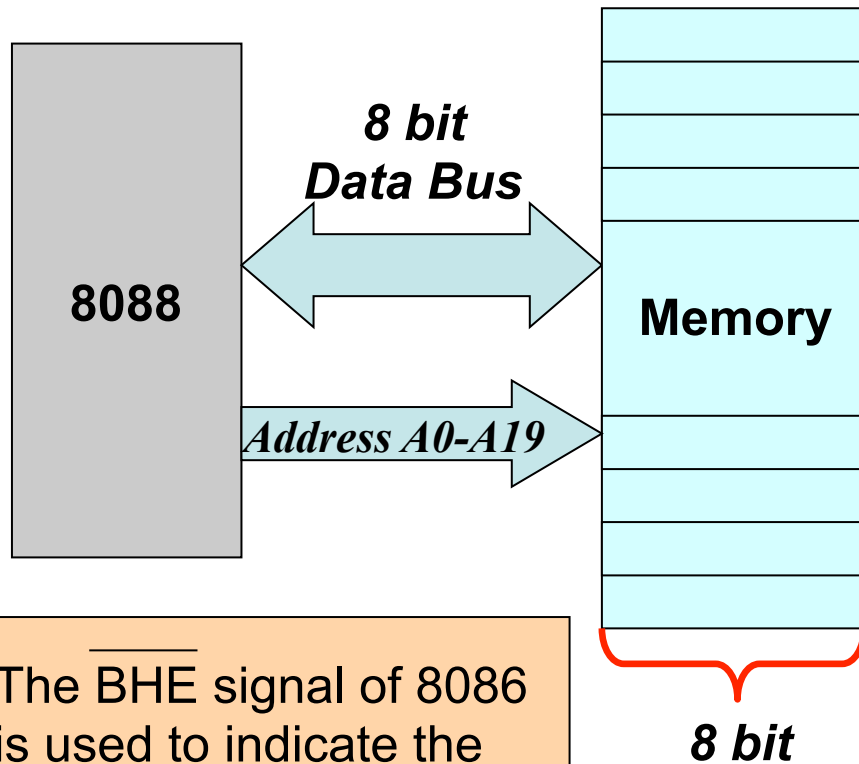
- Module Overview
- Memory and I/O Interfacing
- Interfacing Pins of the Intel 8088  $\mu$ P
- Memory and I/O Addressing
- 8088 vs 8086 Interfacing






# Intel 8086 Microprocessor

- Nearly all the internal functions of 8086 are identical to 8088.
- 8088 uses a 8-bit external data bus, but 8086 uses a 16-bit external data bus.
- In 8086, it has a BHE\ bus high enable signal, which goes low for the data transfers over D15-D8 and is used to select odd address memory bank. (**Take special note of BHE\ signal**)!
- Beware you are asked to design interfaces for 8086 or 8088!

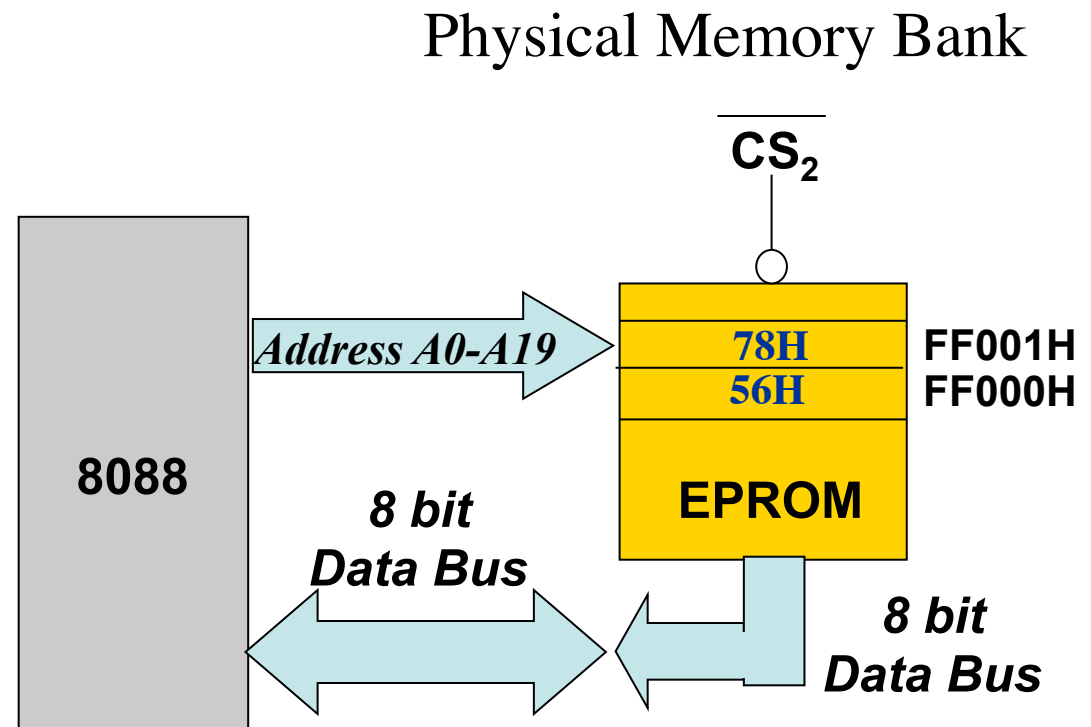
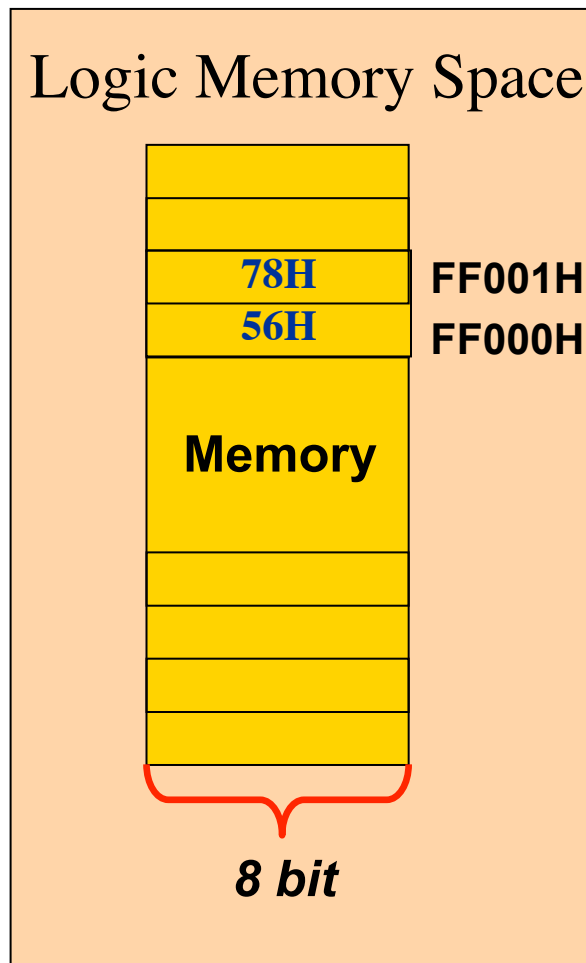
# 8086 Bus High Enable (BHE) Signal



The  $\overline{\text{BHE}}$  signal of 8086 is used to indicate the transfer of data over the higher order (D15-D8) data bus. It is an active low signal meaning it goes to logic low when transferring data over D15-D8.

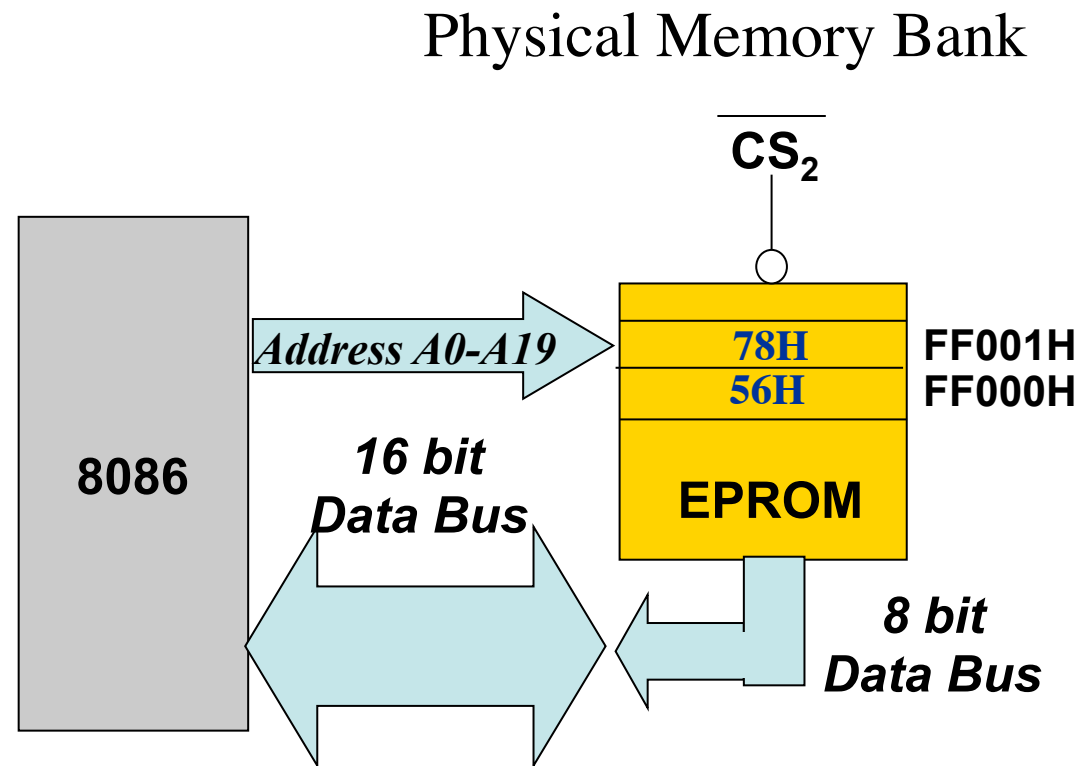
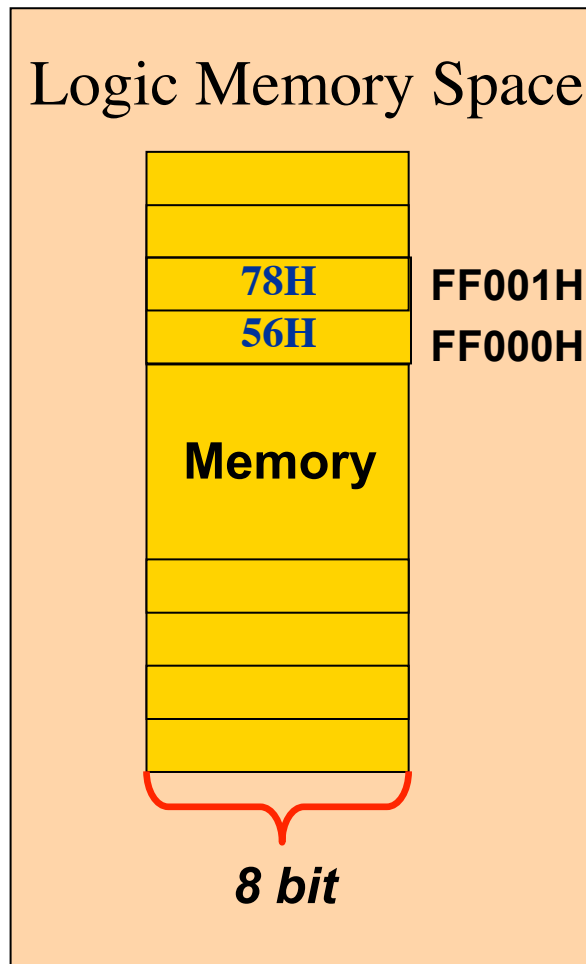
$\overline{\text{BHE}}$	A0	Indication
0	0	Whole Word 
0	1	Upper byte at odd address 
1	0	Lower byte at even address 
1	1	None

# Single-Bank Physical Memory for 8088



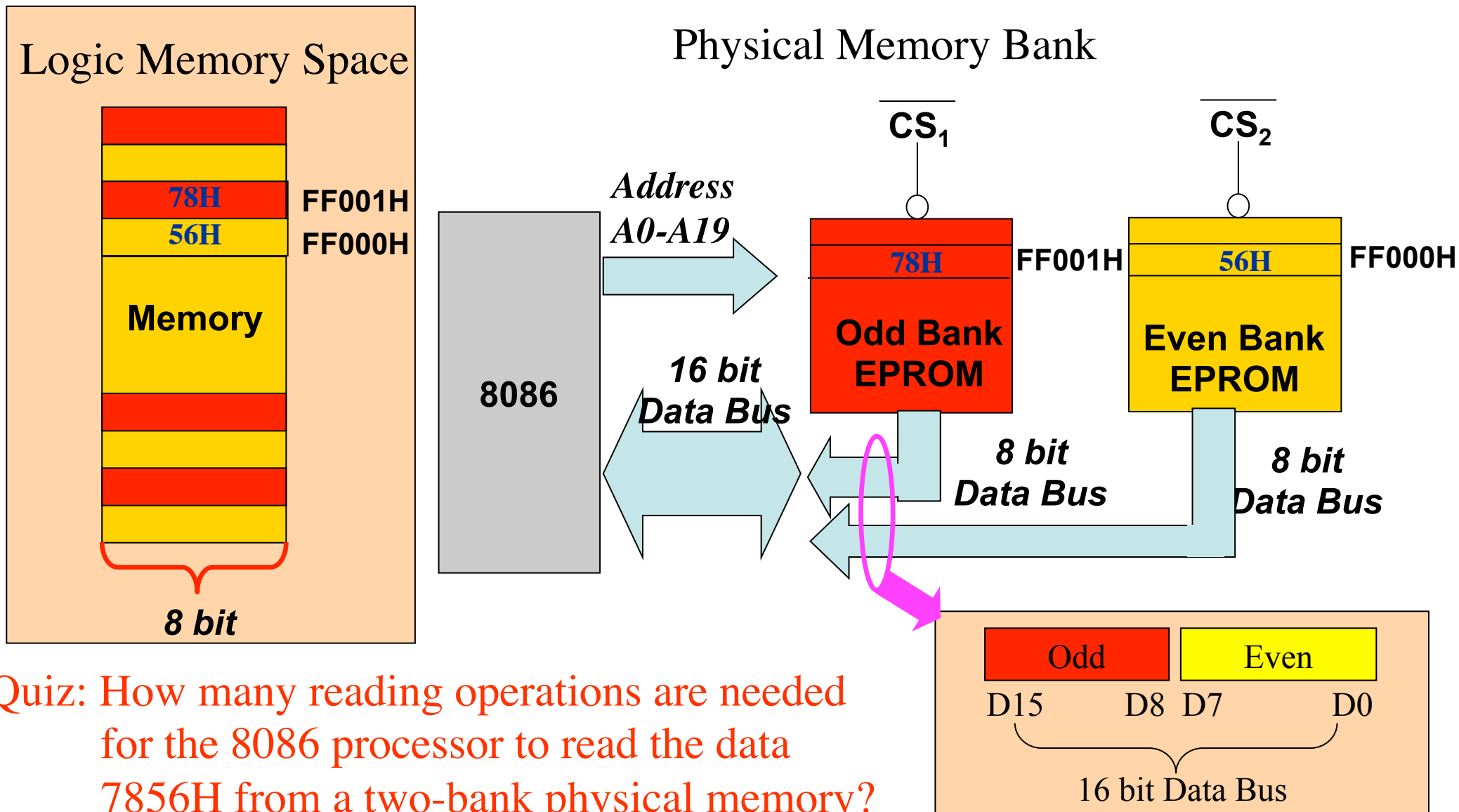
Quiz: How many reading operations are needed for the 8088 processor to read the data 7856H from a single-bank physical memory?

# Single-Bank Physical Memory for 8086



Quiz: How many reading operations are needed for the 8086 processor to read the data 7856H from a single-bank physical memory?

# Two-Bank Physical Memory for 8086



# Relationships Among Electrical level, Logic Level and Active High (Low)

- **Active low (high) signal means that the function of this signal will take effect when the signal receives a logic low (high) signal.**
- **We are using a positive logic system, where a logic low (high) signal is corresponding to the low (high) electrical level signal.**
- **Suppose we have two electrical levels, one is 0V, and the other is +5V.**
- **0V corresponds to the logic low, and it will trigger an active low signal like BHE $\setminus$ .**
- **+5V corresponds to the logic high, and it will trigger an active high signal like HOLD.**

## Sample Problem (1)

**Question :** Interface two 4K \* 8 EPROMS and two 4K \* 8 RAM chips with 8086. Select suitable address maps and find out the logic function for the four chip select signals.

**Solution:** After the reset, the IP and CS are initialized to the address FFFF0H. Hence this address must lie in the EPROM. At least one RAM should be mapped to the address space that covers the interrupt vector table, which occupies the first 1K address space starting from 00000H.

## Sample Problem (2)

Address	A <sub>19</sub>	A <sub>18</sub>	A <sub>17</sub>	A <sub>16</sub>	A <sub>15</sub>	A <sub>14</sub>	A <sub>13</sub>	A <sub>12</sub>	A <sub>11</sub>	A <sub>10</sub>	A <sub>9</sub>	A <sub>8</sub>	A <sub>7</sub>	A <sub>6</sub>	A <sub>5</sub>	A <sub>4</sub>	A <sub>3</sub>	A <sub>2</sub>	A <sub>1</sub>	A <sub>0</sub>
<b>FFFFFFH</b>	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
<b>FE000H</b>	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
<b>01FFFFH</b>	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1
<b>00000H</b>	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### Memory Map for the Problem

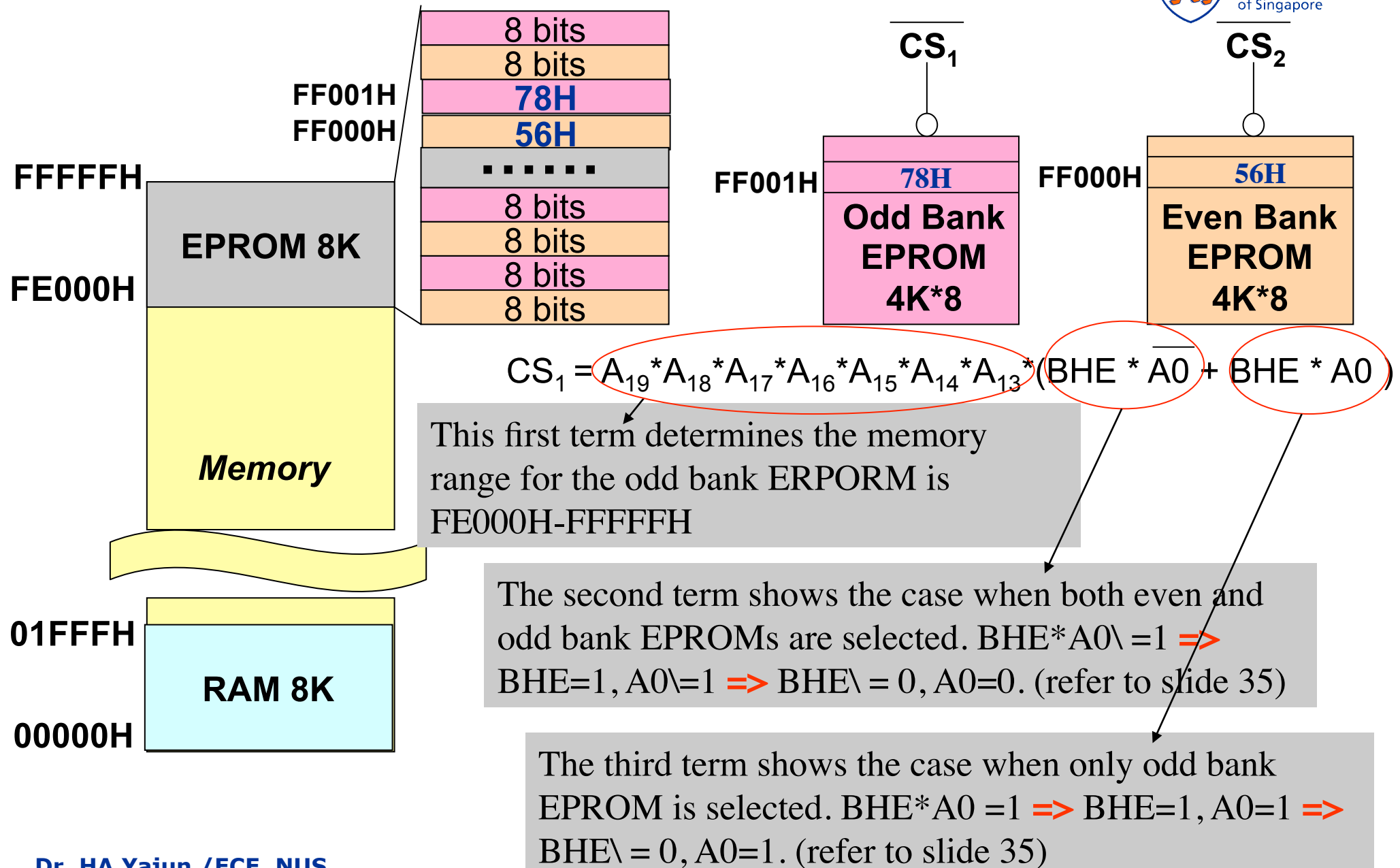
A<sub>0</sub>-A<sub>12</sub> defines a 8K address range.

A<sub>19</sub>-A<sub>13</sub> = 1111111 determines the 8K address range is FE000H – FFFFFFFH.

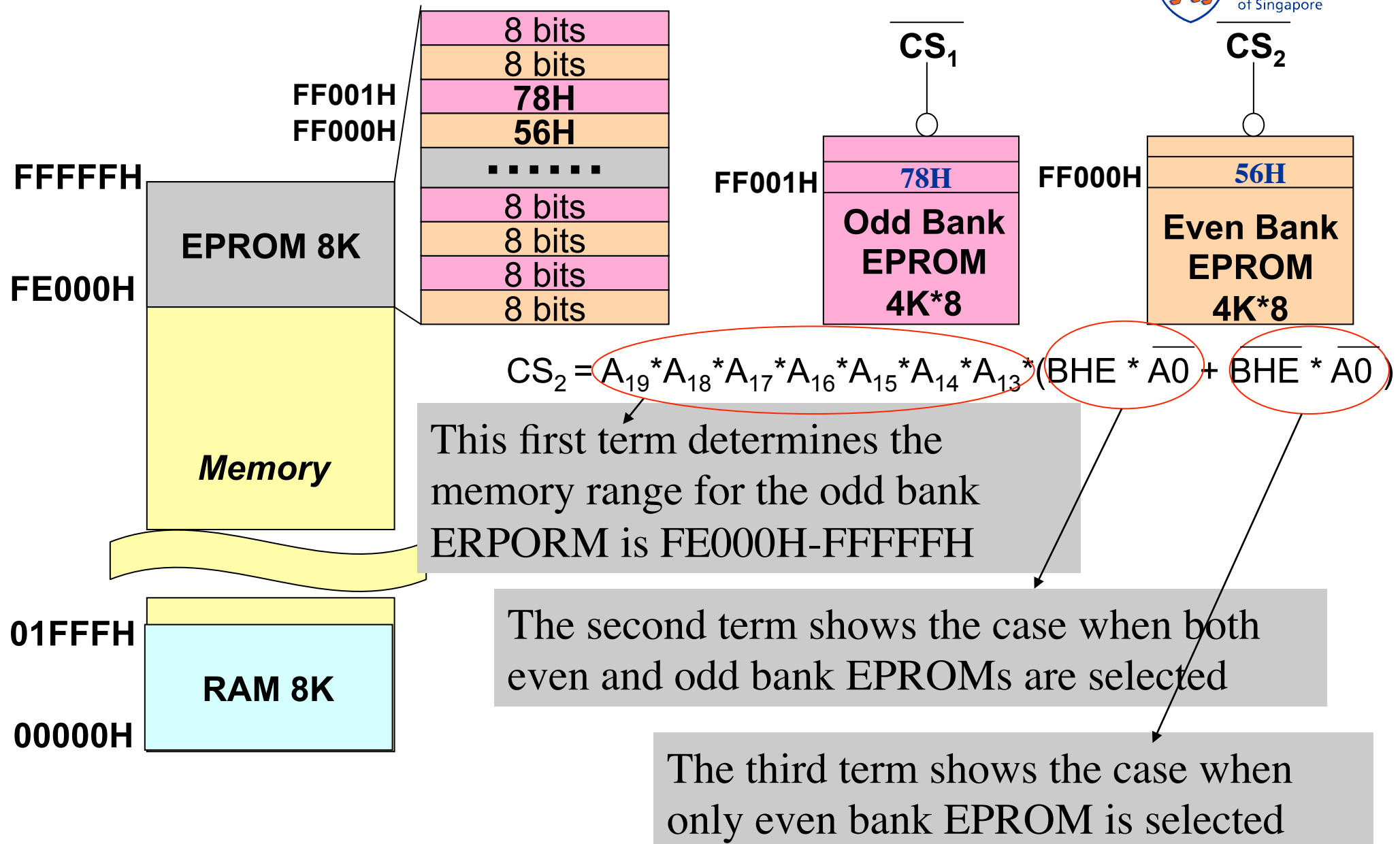
A<sub>19</sub>-A<sub>13</sub> = 0000000 determines the 8K address range is 00000H – 01FFFFH.



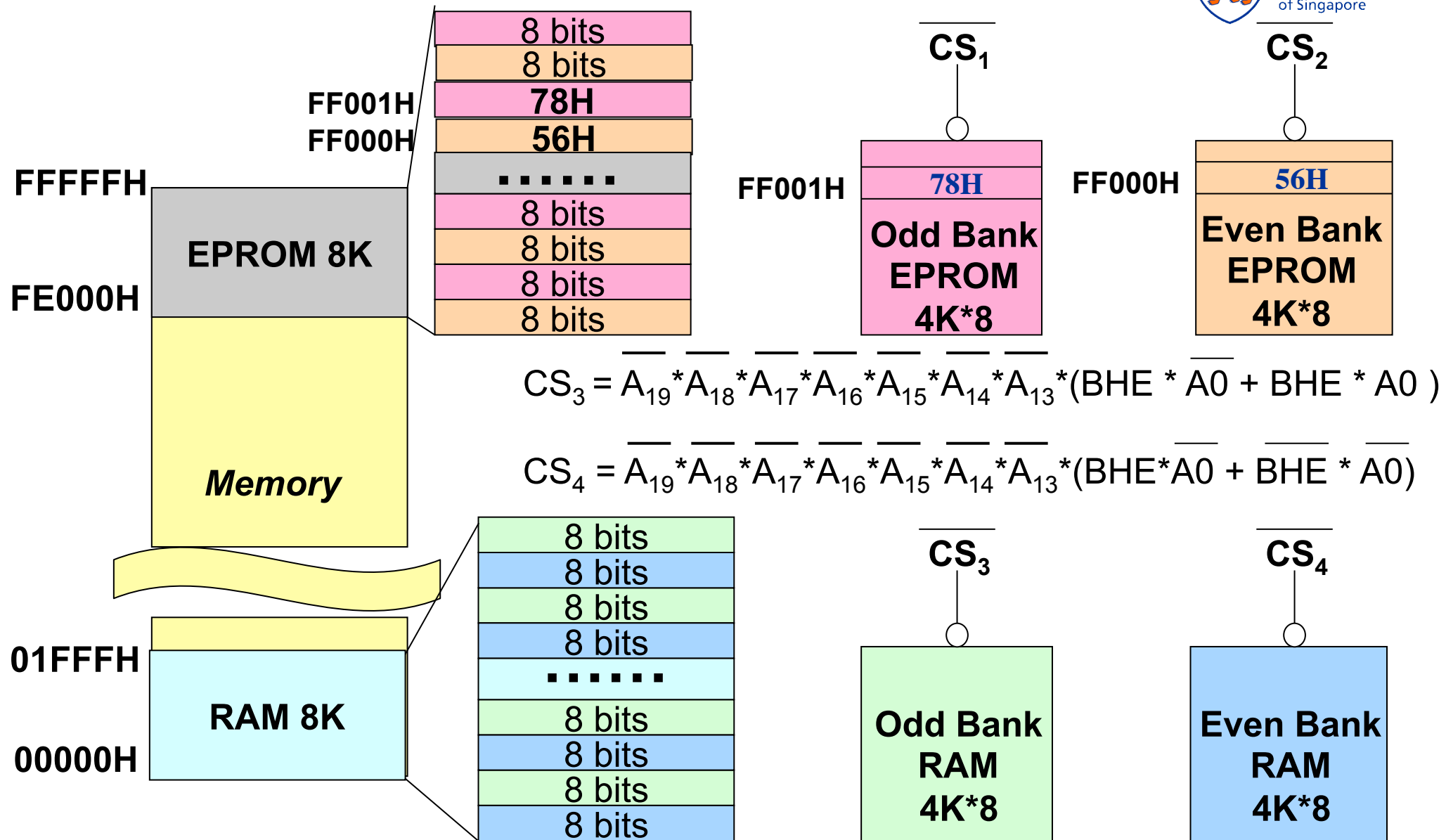
# Sample Problem (3)



# Sample Problem (4)




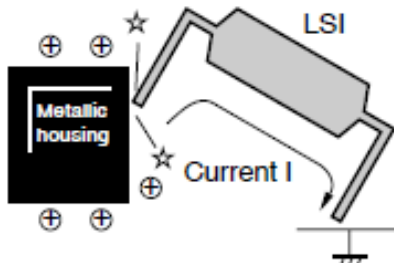
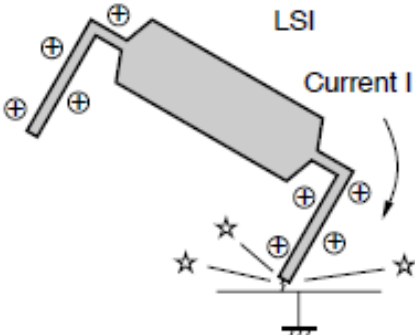
# Sample Problem (5)



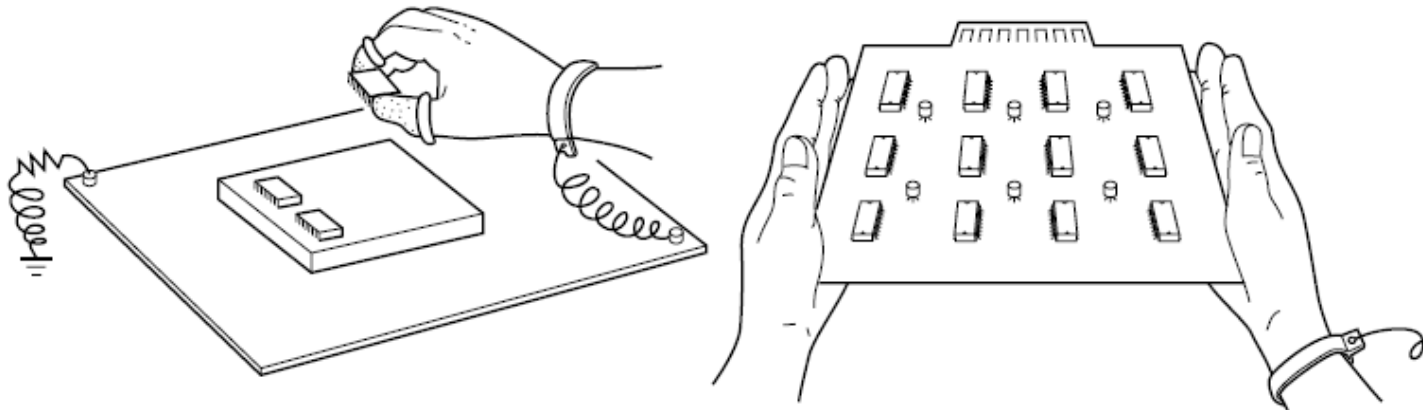
# Major Project Debugging Guide

- **Avoid ESD to damage your semiconductor devices.**
- **Use the correct wire wrapping way to have robust wirings.**
- **Before switching on power, always check if VDD and ground has been shorted.**
- **Build and test circuits block by block, **not in only one step!****
- **First to make sure that power and clock are working fine.**
- **Second to make sure CPU and RAM/ROM chips are working.**
- **Third to make sure 8255 chip is working fine.**

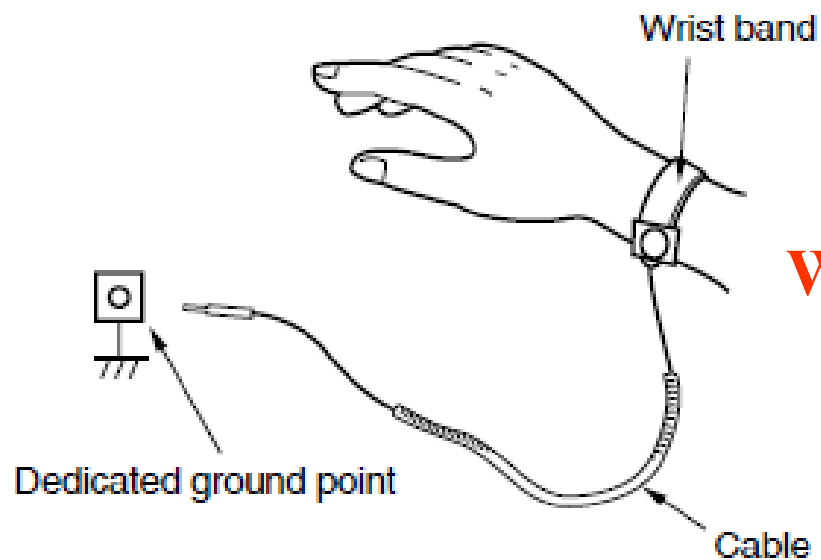
# ElectroStatic Discharge (ESD) Damages Devices

Model Name	Description	Figure (Example) Note	Cause (Example)
Human Body Model (HBM)	Charge built up on the human body is discharged when a part of the body touches a pin of the device. The device is damaged by discharged current if a pin of the device is grounded.		<ul style="list-style-type: none"> <li>• Not wearing wrist strap and conductive shoes</li> <li>• Wearing clothing which readily generates static electricity</li> <li>• Directly touching pins with the hand</li> </ul>
Machine Model (MM)	Charge built up on a metallic object is discharged when the object touches a pin of the device. The device is damaged by discharged current if a pin of the device is grounded.		<ul style="list-style-type: none"> <li>• Current leaking from soldering iron</li> <li>• Leakage current of equipment, insufficient grounding, etc.</li> </ul>
Charged Device Model (CDM)	The conductors (such as the chip, wires, and lead frames) of the device are charged, and discharging takes place when a pin of the device touches equipment, a jig, or a tool. The charge built up on the conductor of the device is discharged as soon as the device is grounded, damaging the device.		<ul style="list-style-type: none"> <li>• Contact of charged object with pin of device</li> <li>• Dropping or touching charged device on a metallic object</li> </ul>

# Ways to Avoid ESD

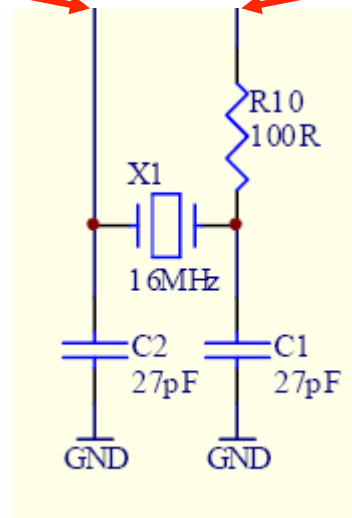
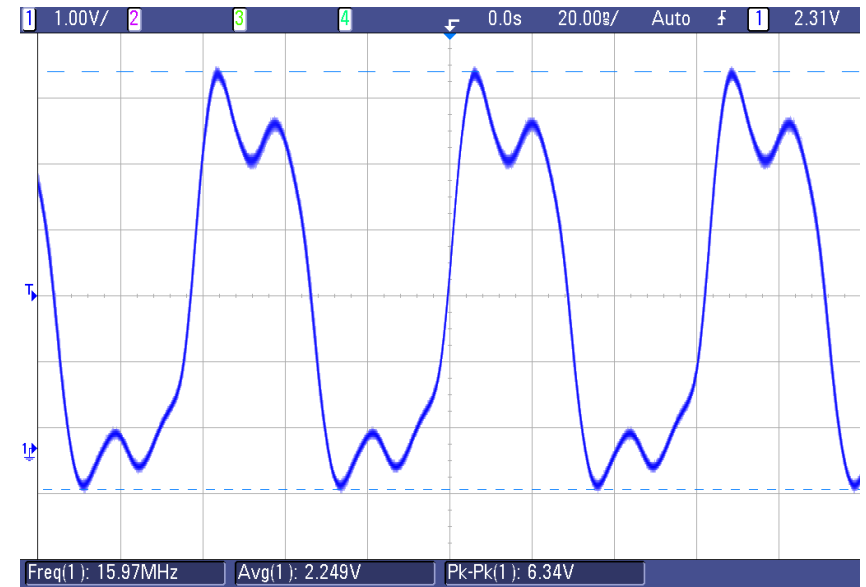
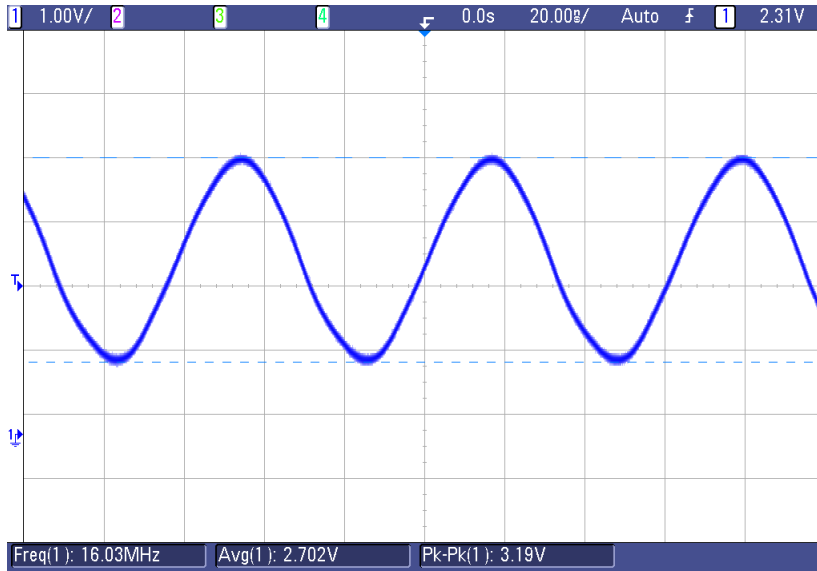


**Do not touch the pins of devices and printed wiring boards!**



**Wear a wrist strap if possible!**

# Waveforms from Clock Generation Circuit



# Waveforms from the Data Bus

After constructing the circuit with RAM and ROM (but without 82'55 PPI chip), you can load the EEPROM with your program. Observe the waveforms of the data-bus using the oscilloscope. If your program is able to transfer data between the CPU and ROM, you should be able to see, from the data-bus, similar waveforms as shown in the figure above. The point is that the CPU is continuously transferring data from/to the ROM, and therefore you should observe that the waveform keeps changing (if you are transferring different data).

