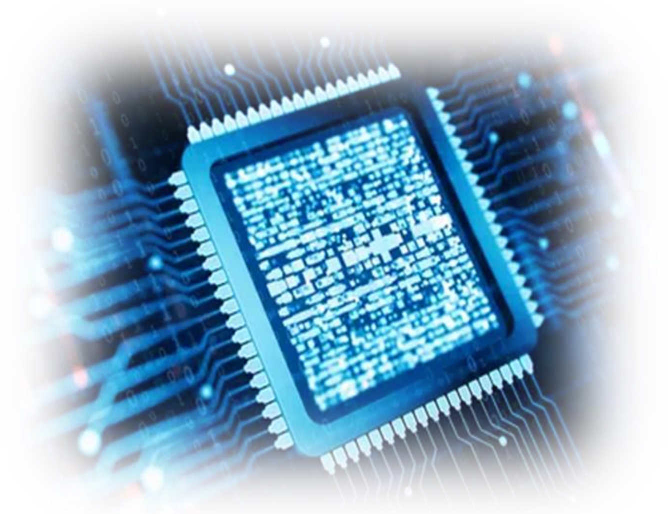


MICROPROCESSOR SYSTEMS

BLG212E

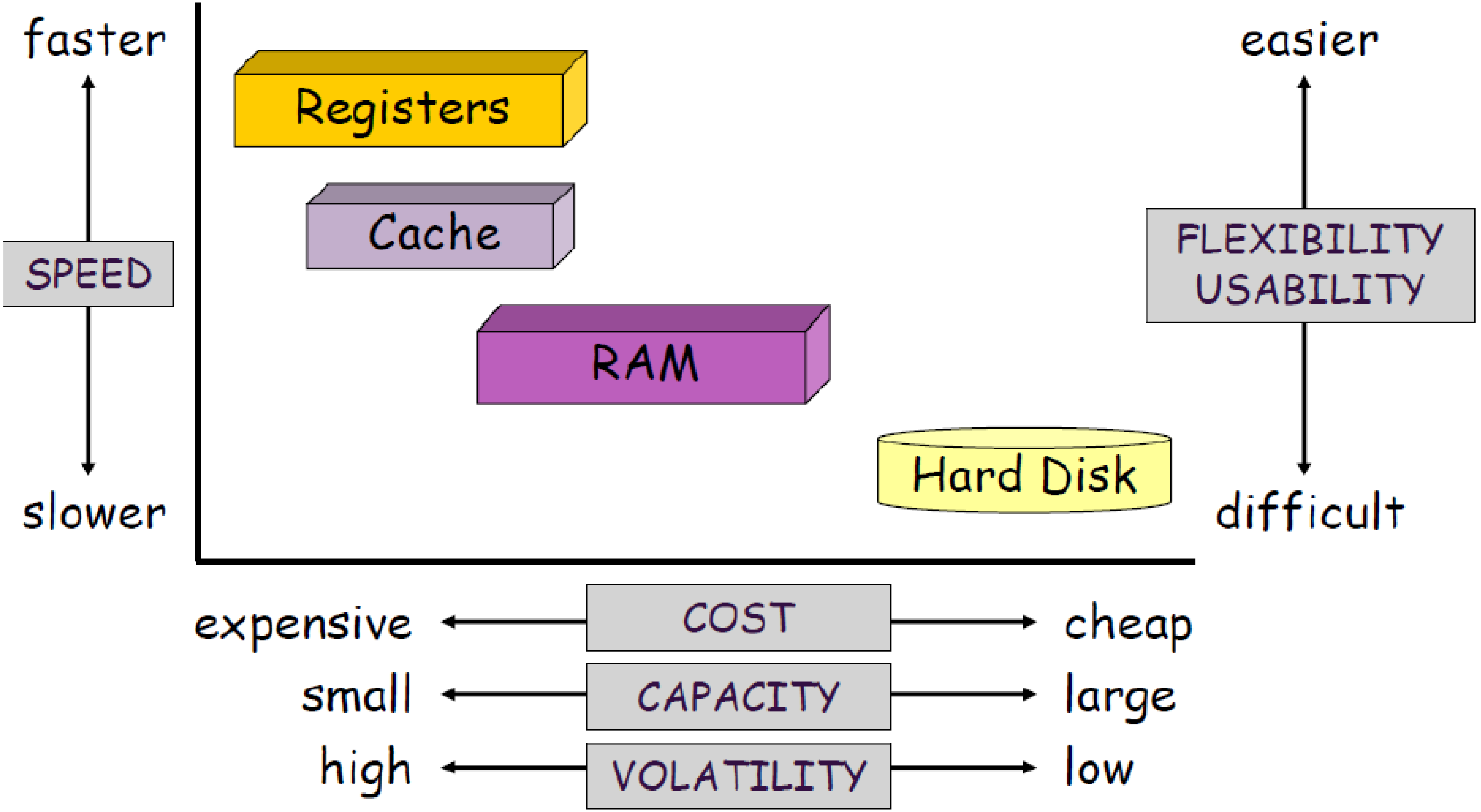
Burak Berk Üstündağ CRN: 11450

Gökhan İnce / Ayşe Yılmazer CRN: 11446

Faculty of Computer and Informatics Engineering
Istanbul Technical University

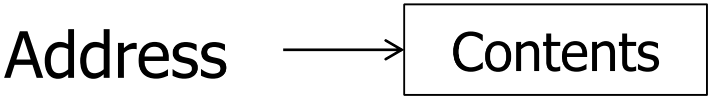
Week 7: Addressing and Memory Organization in Computers

Comparison of Memory Modules



Memory Addressing

- Memory consists of a sequence of directly addressable "locations".
- A memory location is referred to as an information unit.
- An information unit has two components:
  - address
  - contents



Address

Contents

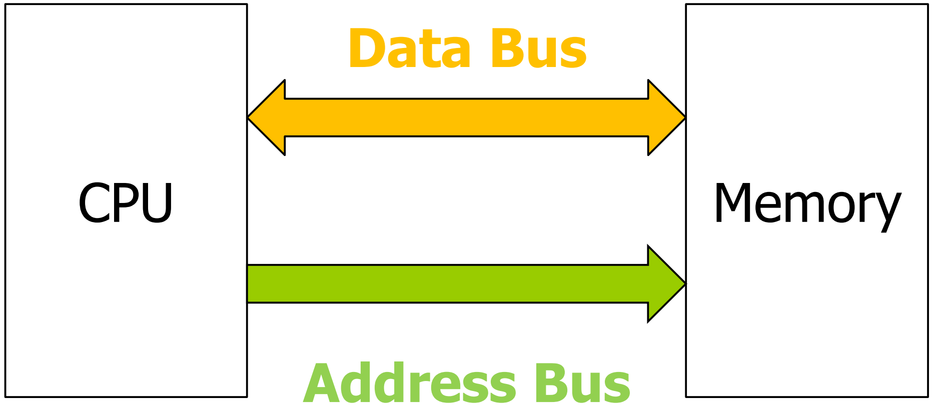
Memory Addressing

- Each location in memory has an address that must be supplied before its contents can be accessed.

- The CPU communicates with memory by first identifying the location's address and then passing this address on the address bus.

- The data are transferred between memory and the CPU along the data bus.

- The number of bits that can be transferred on the data bus at once is called the data bus width of the processor.



Dimensions of Memory

- Memory is usually measured by two numbers: its length and its width (length x width).
  - The length is the total number of locations.
  - The width is the number of bits in each location.
- The length (total number of locations) is a function of the number of address lines.

  # of memory locations = $2^{(\text{\# of address lines})}$

  - A memory chip with 10 address lines would have

    2^{10} = 1024 locations (1K)
  - A memory chip with 4K locations would need

    $\log_2 4096 = 12$ address lines

Educational CPU and Memory

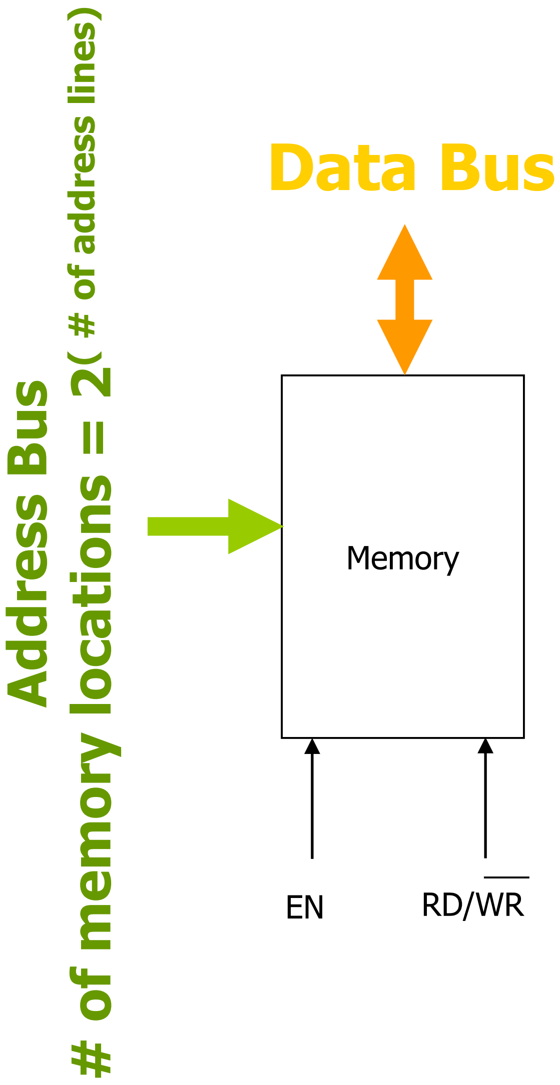
- Educational CPU has 16 address lines. That means it can address

  2^{16} = 64K memory locations.

- Then it will need 1 memory chip with 64 K locations, or 2 chips with 32 K in each, or 4 with 16 K each or 16 of the 4 K chips, etc.
- How would we use these address lines to control the multiple chips?

Chip Select

- Usually, each memory chip has a **Chip Select (CS) input**. The chip will only work if an active signal is applied on that input.
- To allow the use of **multiple chips** in the make up of memory, we need to use a number of the address lines for the purpose of **chip selection**.
- These address lines are decoded to generate the 2^n necessary CS inputs for the memory chips to be used.

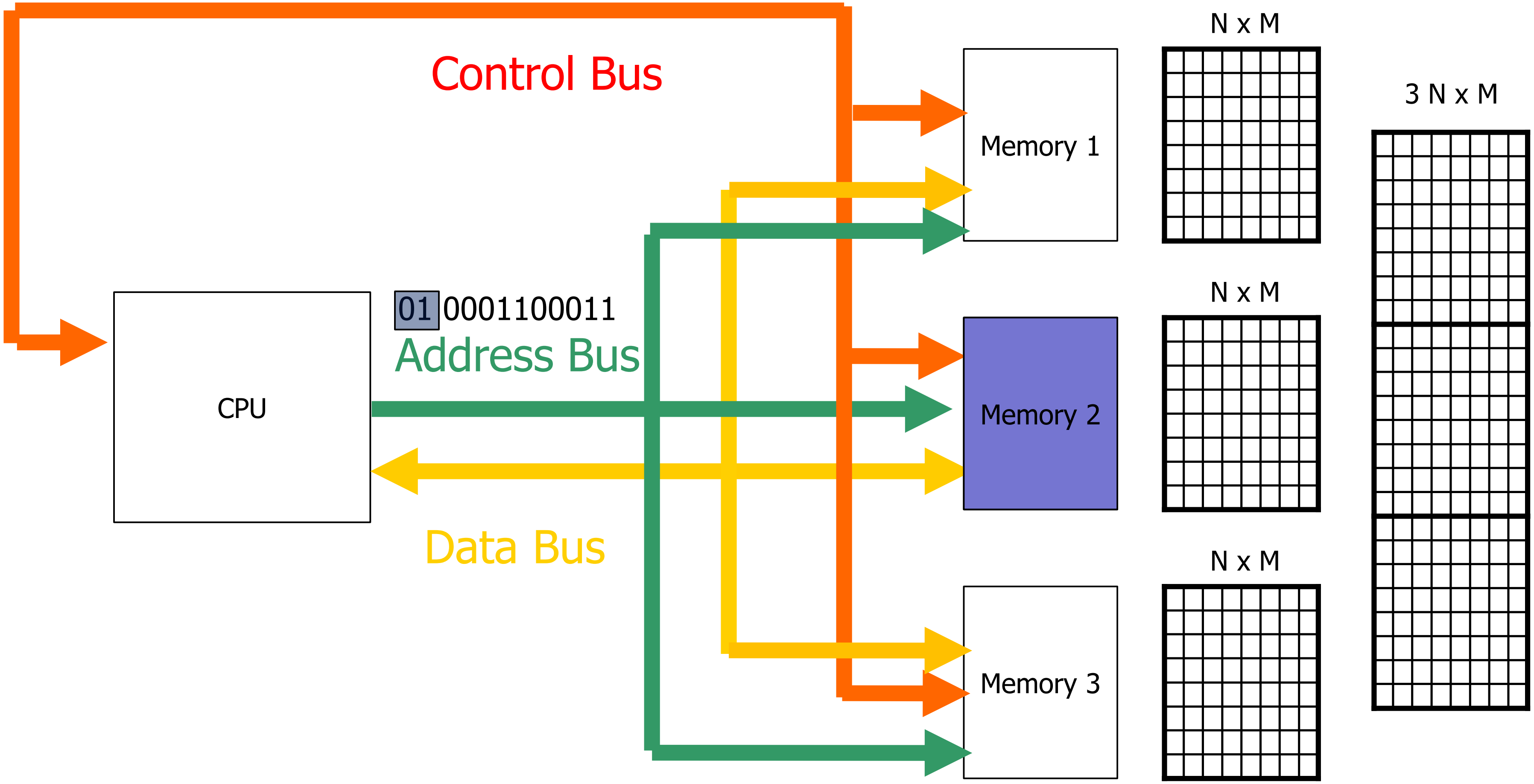


Data Bus

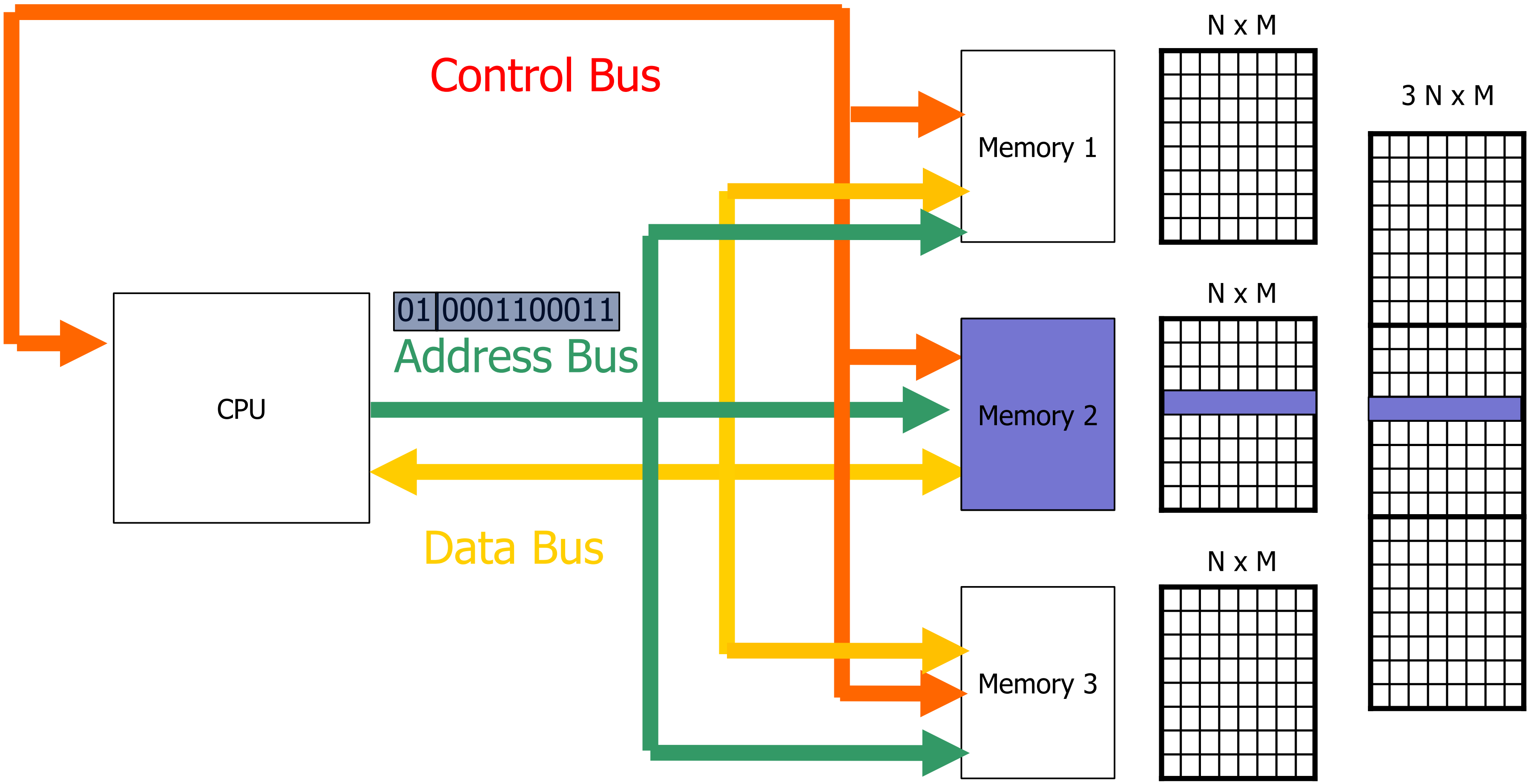
Address Bus

# of memory locations = 2(# of address lines)

Memory Access

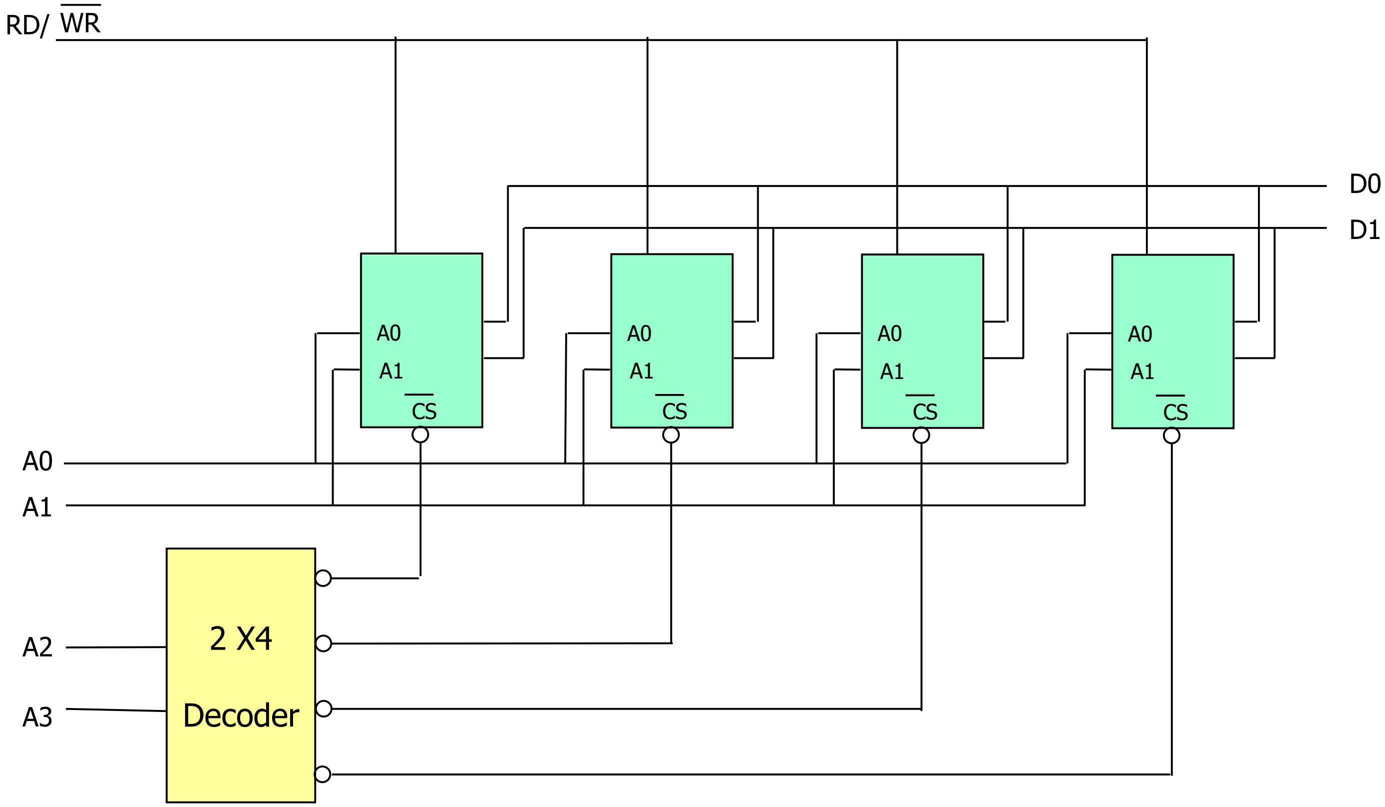


Memory Access



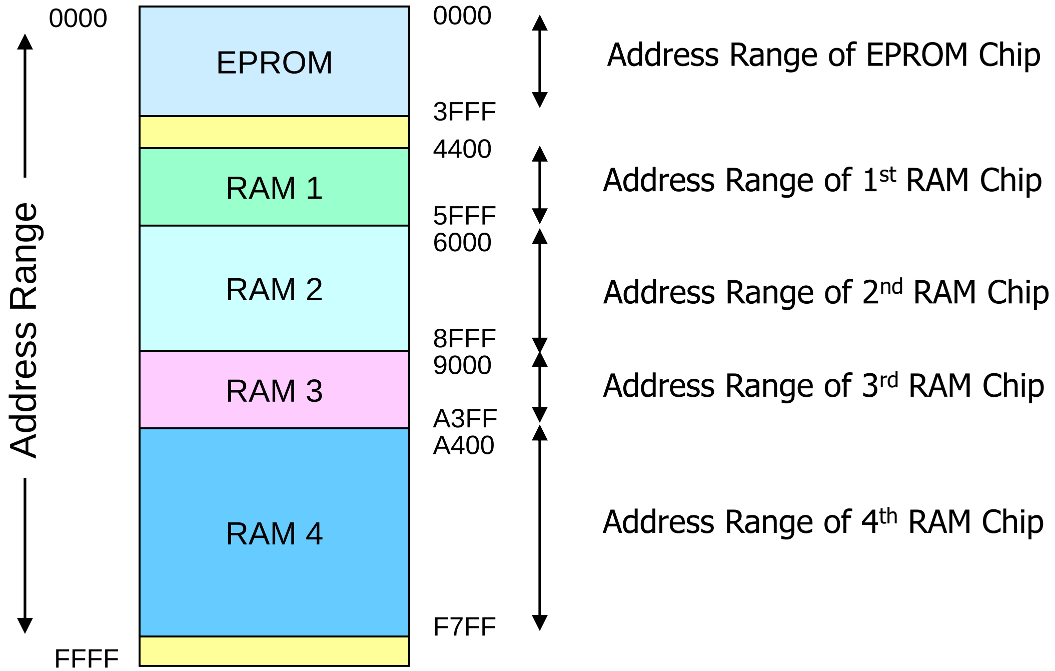
Chip Selection Example

A memory system made up of 4 of the 4x2 memory chips



Memory Map

- Designates the address space for each memory chip



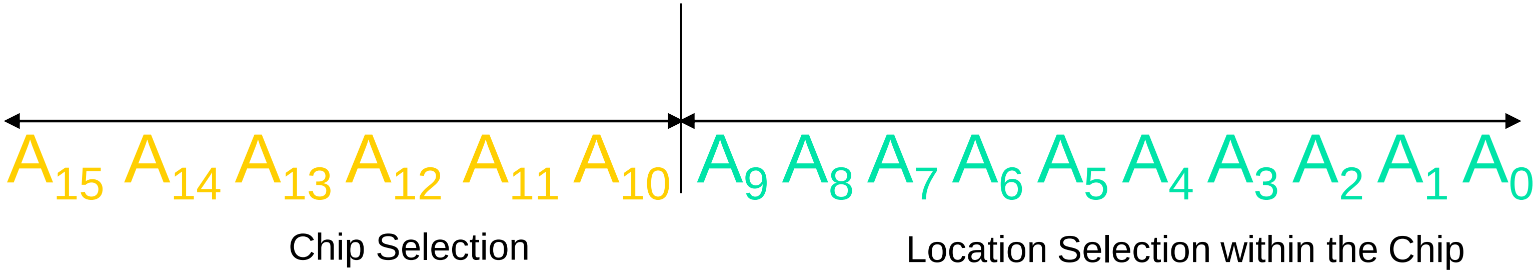
Address Range
0000
FFFF
EPROM
RAM 1
RAM 2
RAM 3
RAM 4
0000
3FFF
4400
5FFF
6000
8FFF
9000
A3FF
A400
F7FF
Address Range of EPROM Chip
Address Range of 1st RAM Chip
Address Range of 2nd RAM Chip
Address Range of 3rd RAM Chip
Address Range of 4th RAM Chip

Address Range of a Memory Chip

- The **address range of a particular chip** is the list of all addresses that are mapped to the chip.
- An 8-bit CPU with **16 address bits** can address a total of **64K memory locations**.
  - If we use memory chips with 1K locations each, then we will need 64 such chips.
  - The 1K memory chip needs 10 address lines to uniquely identify the 1K locations. ($\log_2 1024 = 10$)
  - That leaves 6 address lines which is the exact number needed for selecting between the 64 different chips ($\log_2 64 = 6$).

Address Range of a Memory Chip

- 16 bit address lines can be separated into two pieces



A_{15} A_{14} A_{13} A_{12} A_{11} A_{10} | A_9 A_8 A_7 A_6 A_5 A_4 A_3 A_2 A_1 A_0

Chip Selection

Location Selection within the Chip

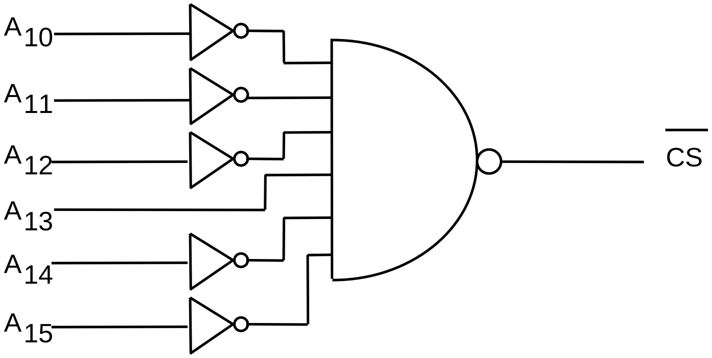
  - Depending on the combination on the address lines A_{15}-A_{10}, the address range of the specified chip is determined.

Chip Select Example

- A chip that uses the combination [A_{15}-A_{10}]= 001000 would have addresses that range from \$2000 to \$23FF.
  - The 10 address lines on the chip gives a range of

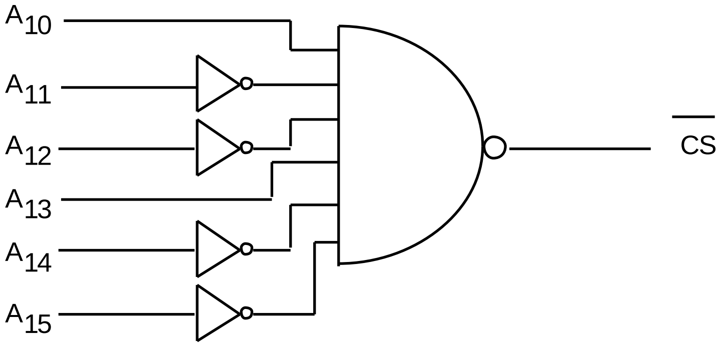
    xxxx xx00 0000 0000 to xxxx xx11 1111 1111 or

    \$x000 to \$x3FF for each of the chips.
  - The memory chip in this example would require the following NAND circuit on its chip select input:



Chip Select Example

- If we change the above combination to the following:

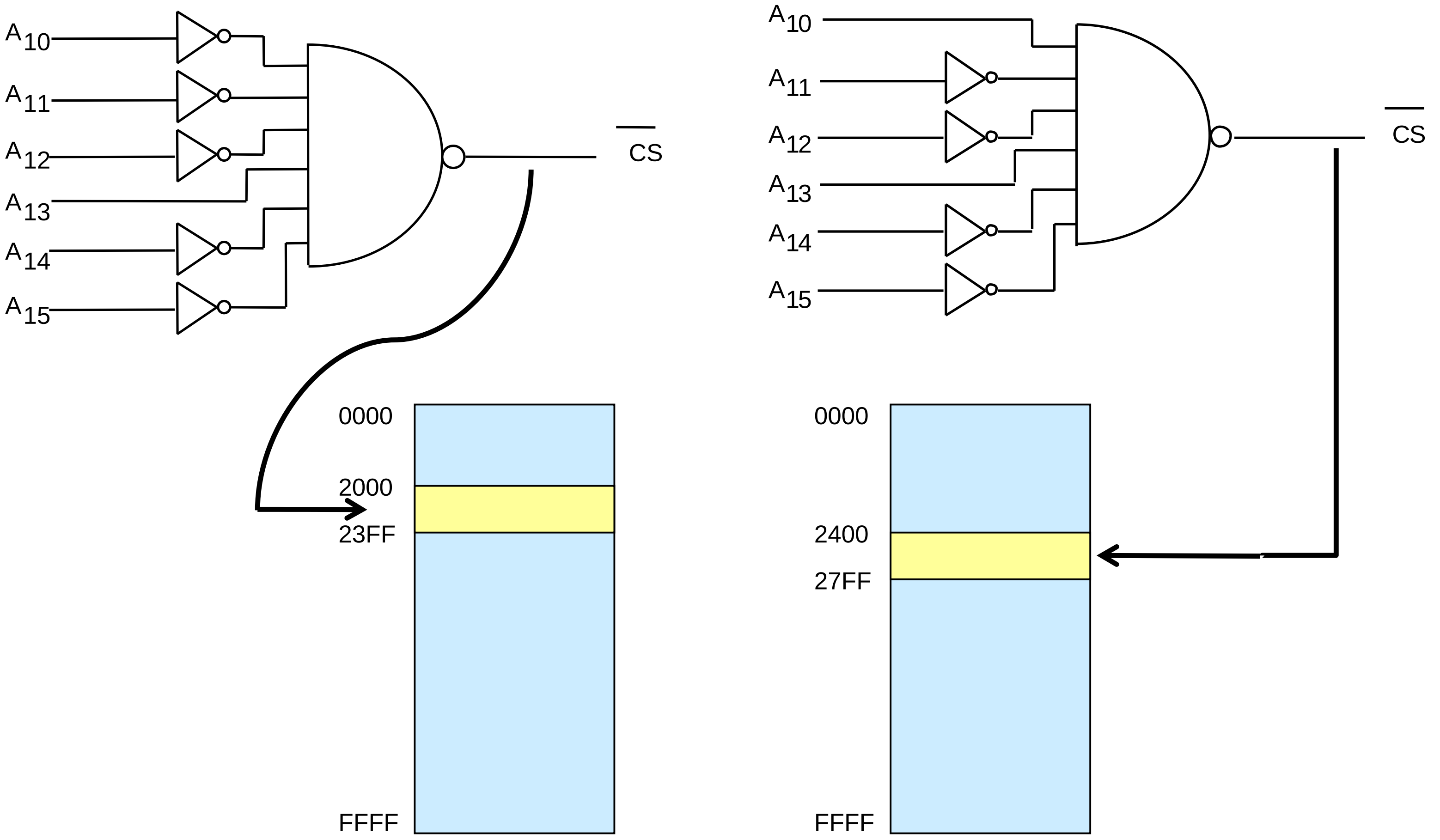


  - Now the chip would have addresses ranging from: 2400 to 27FF.

A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
0	0	1	0	0	1	0	0	0	0	0	0	0	0	0	0	=>$2400
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	=>$27FF

  - Changing the combination of the address bits connected to the chip select changes the address range for the memory chip.

Chip Select Example



Memory Organization

- Example: For a CPU with 8-bit data bus and 16-bit address bus, build the memory that spans between \$0000 and \$1FFF with 2Kx8 memory chips.

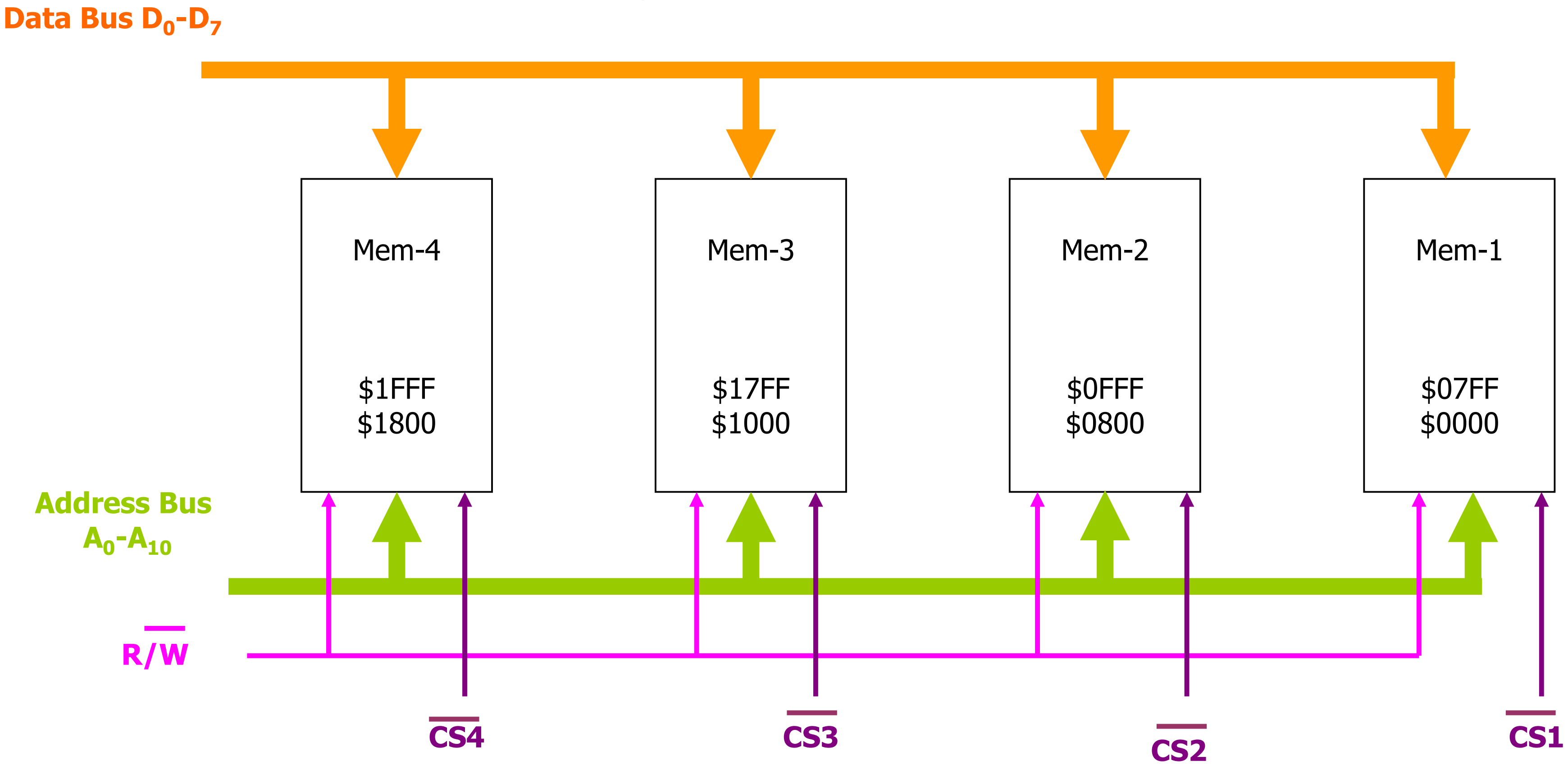
- What is the required memory space?
- How many 2K chips are needed?

Memory Organization

$A_{15}A_{14}A_{13}A_{12}$	$A_{11}A_{10}A_9A_8$	$A_7A_6A_5A_4$	$A_3A_2A_1A_0$	
0000	0000	0000	0000	$0000
0000	0111	1111	1111	$07FF
0000	1000	0000	0000	$0800
0000	1111	1111	1111	$0FFF
0001	0000	0000	0000	$1000
0001	0111	1111	1111	$17FF
0001	1000	0000	0000	$1800
0001	1111	1111	1111	$1FFF

Memory Organization

- Connect the DATA BUS, ADDRESS BUS, R/W together
- CS is determined using A12 and A11



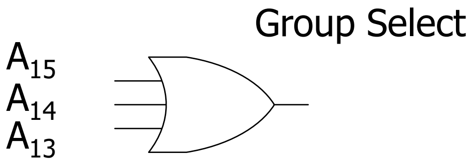
Memory Organization

- Chip select is done with address bits that are not used within the memory chip.

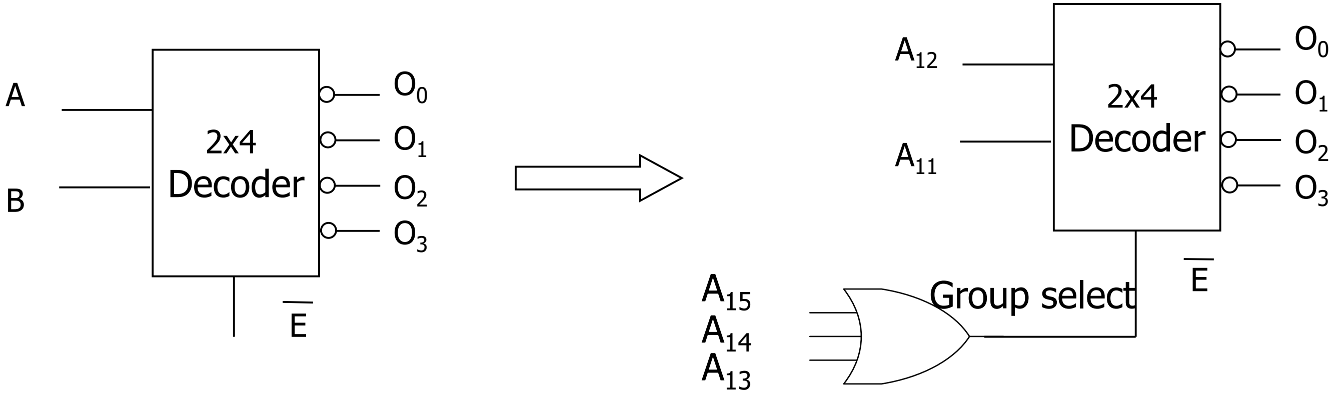
	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}.....
Memory 1	0	0	0	0	0	Used to address locations within a memory chip
Memory 2	0	0	0	0	1	
Memory 3	0	0	0	1	0	
Memory 4	0	0	0	1	1	

Memory Organization

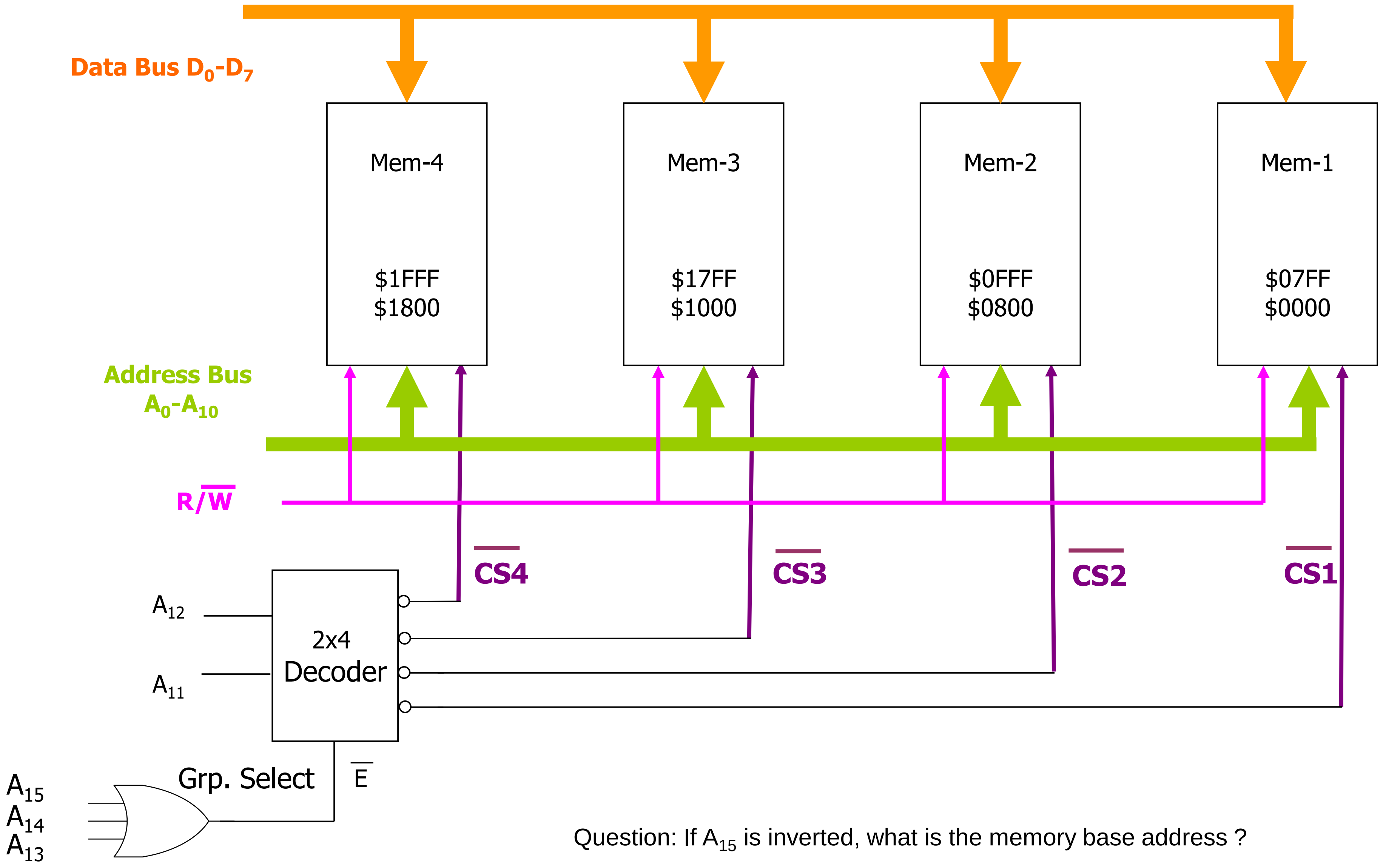
- A_{15}, A_{14}, A_{13} remain at low at all times. They can be used to form another Chip Select (Group Select)



- A_{12} and A_{11} can be used to select memory chips with 2x4 decoder

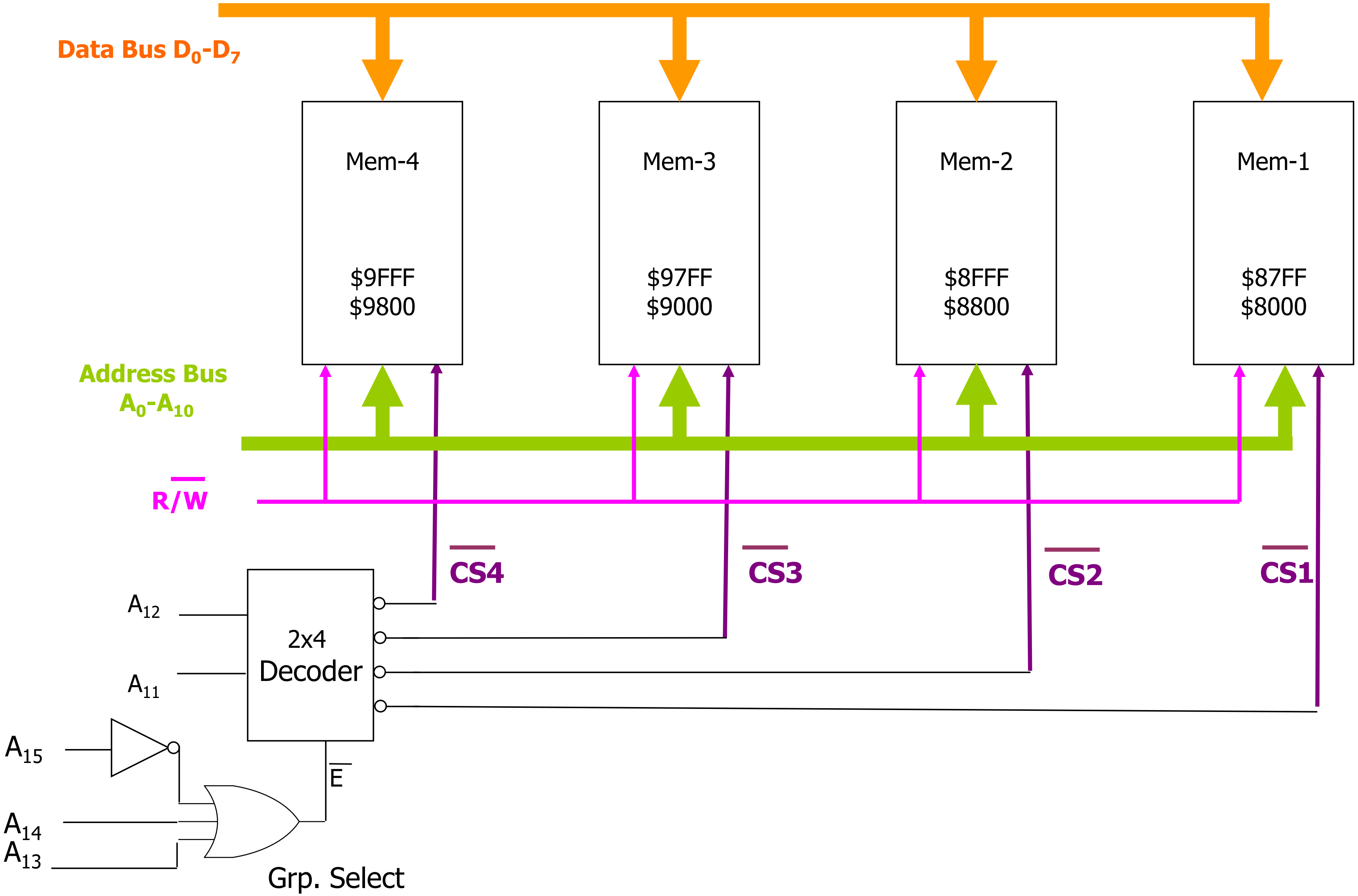


Memory Organization



Question: If A_{15} is inverted, what is the memory base address ?

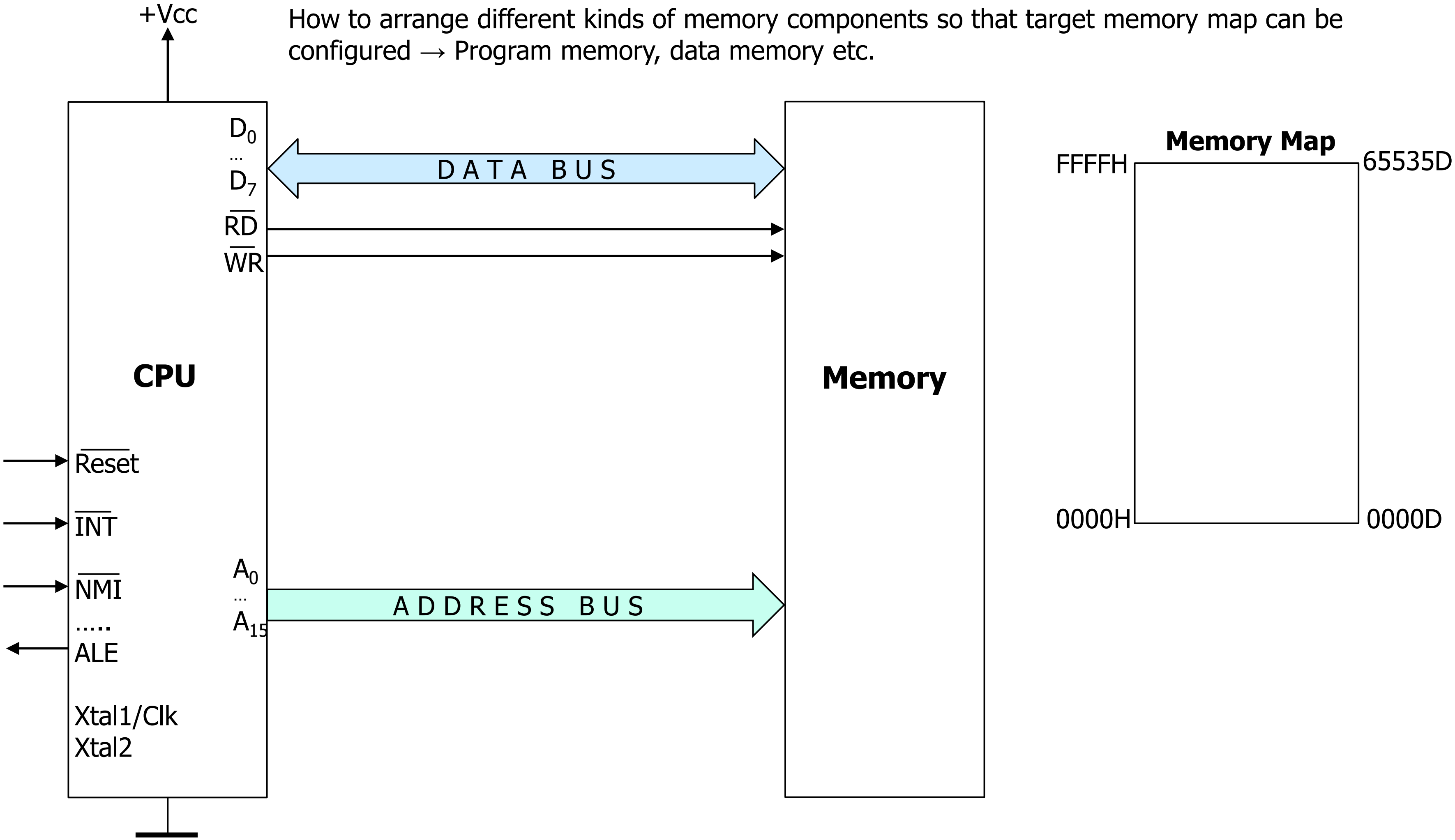
Memory Organization



Memory Map (Example 1)

CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H

Problem:
How to arrange different kinds of memory components so that target memory map can be configured → Program memory, data memory etc.



Memory Map

CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H, 32K EPROM, 16K SRAM, 8K EEPROM

1- 32K EPROM : 32x8kbit → 27C256: P, $\overline{OE}$, $\overline{CE}$

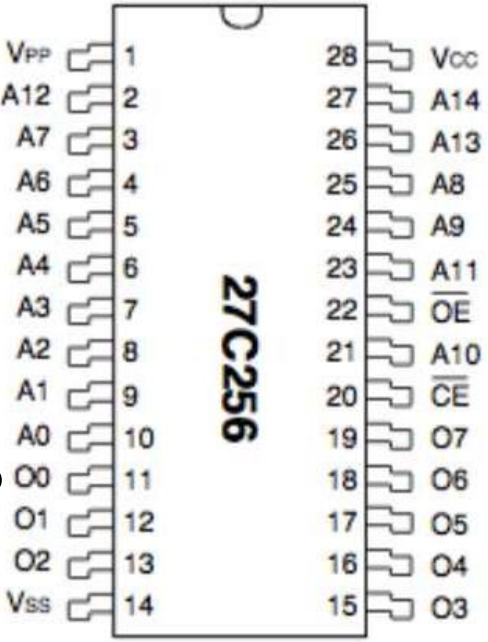
2- 27C→$O_0…O_7$, Generic code: 27C256

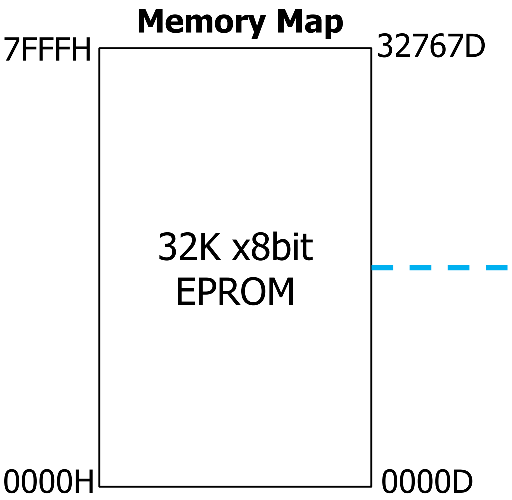
3- x256→ M=256kbit=256x1024 bit

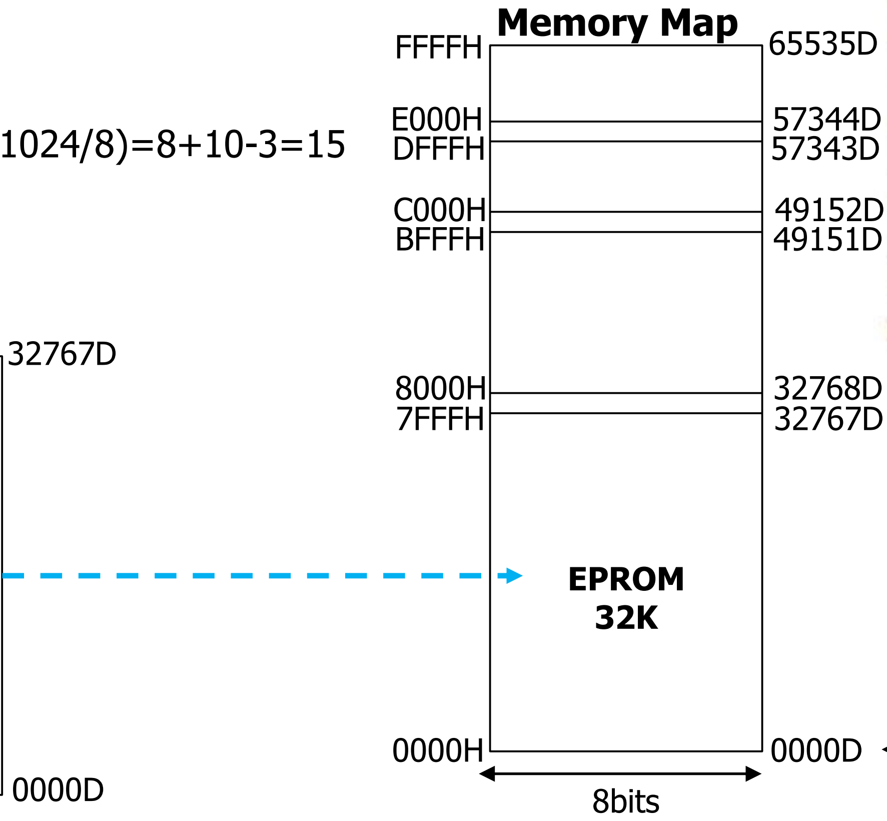
Since 27C refers 8 bits I/O,

Address bus size= $\log_2$ (256x1024/8)=8+10-3=15

Address lines→ A0..A14







Reset Vector

Memory Map

CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H, 32K EPROM, 16K SRAM, 8K EEPROM

1- 16K SRAM : 16x8kbit → 62C128: $\overline{WE}$, $\overline{OE}$, $\overline{CE}$

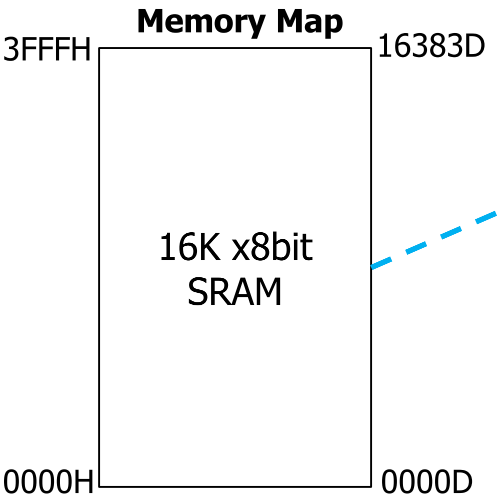
2- 62C→$I/O_0 \ldots I/O_7$, Generic code: 62C128 alternative: **W24129A**

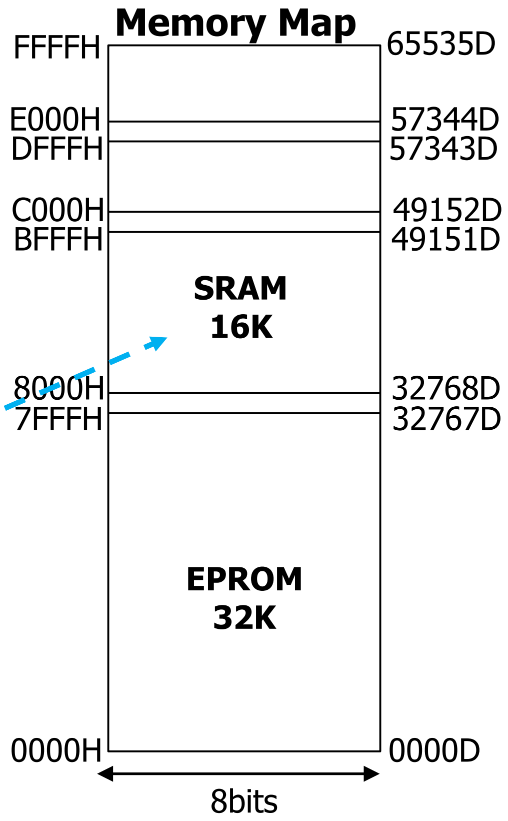
3- x128→ M=128kbit=128x1024 bit

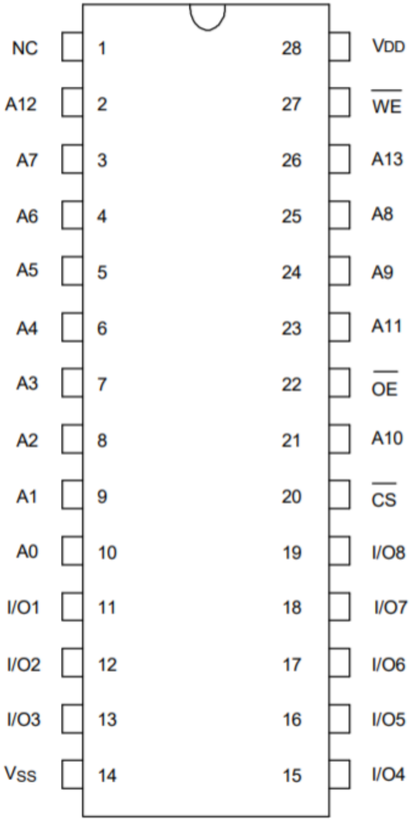
Since 62C/M48 refers 8 bits I/O,

Address bus size= $\log_2$ (128x1024/8)=7+10-3=14

Address lines→ A0..A13







Memory Map

CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H, 32K EPROM, 16K SRAM, 8K EEPROM

1- 8K EEPROM : 8x8kbit → 28C64: $\overline{WE}$, $\overline{OE}$, $\overline{CE}$

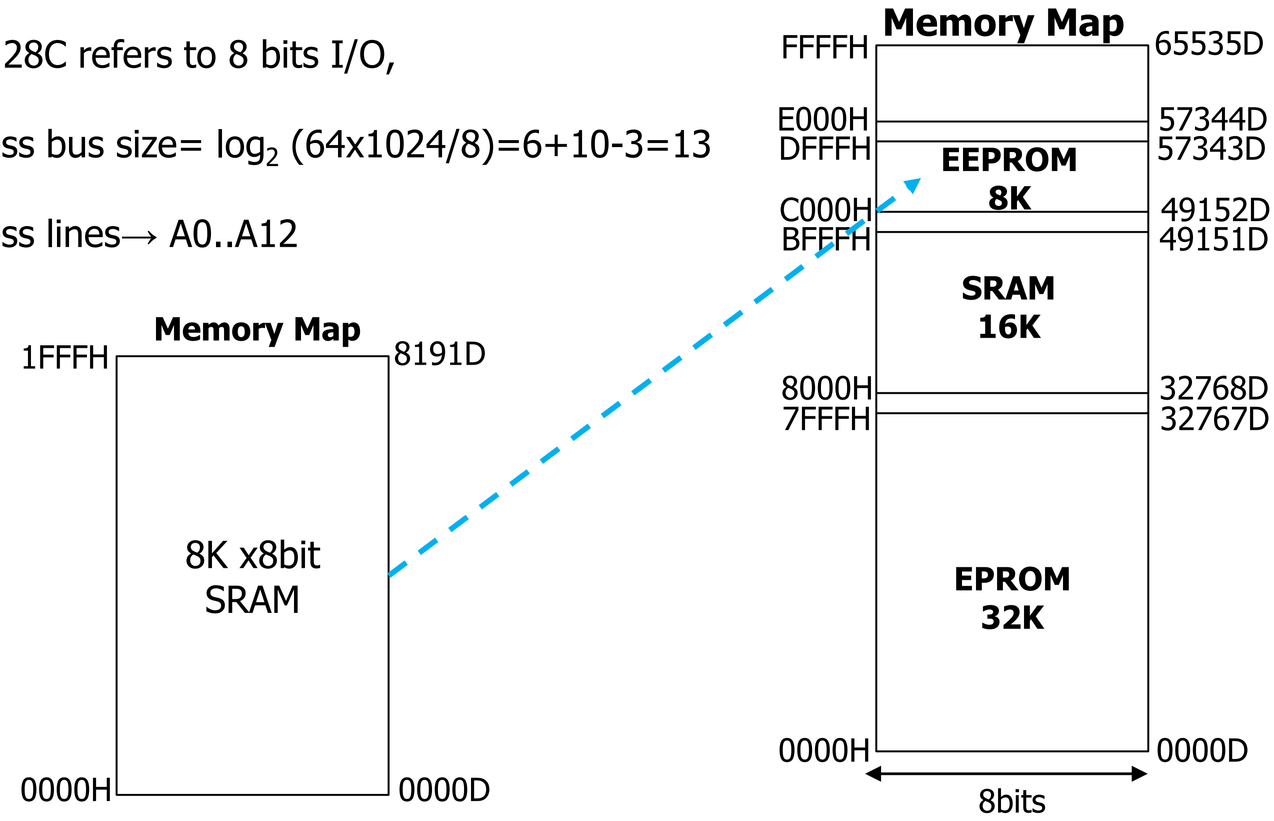
2- 28C→$I/O_0 \ldots I/O_7$, Generic code: 28C64

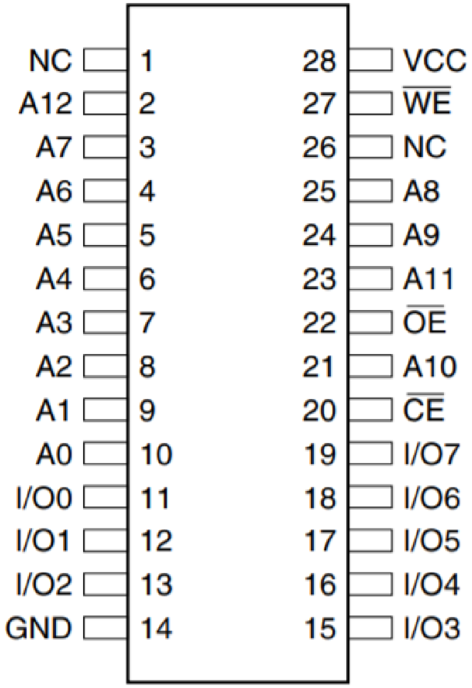
3- x64→ M=64kbit=64x1024 bit

Since 28C refers to 8 bits I/O,

Address bus size= $\log_2$ (64x1024/8)=6+10-3=13

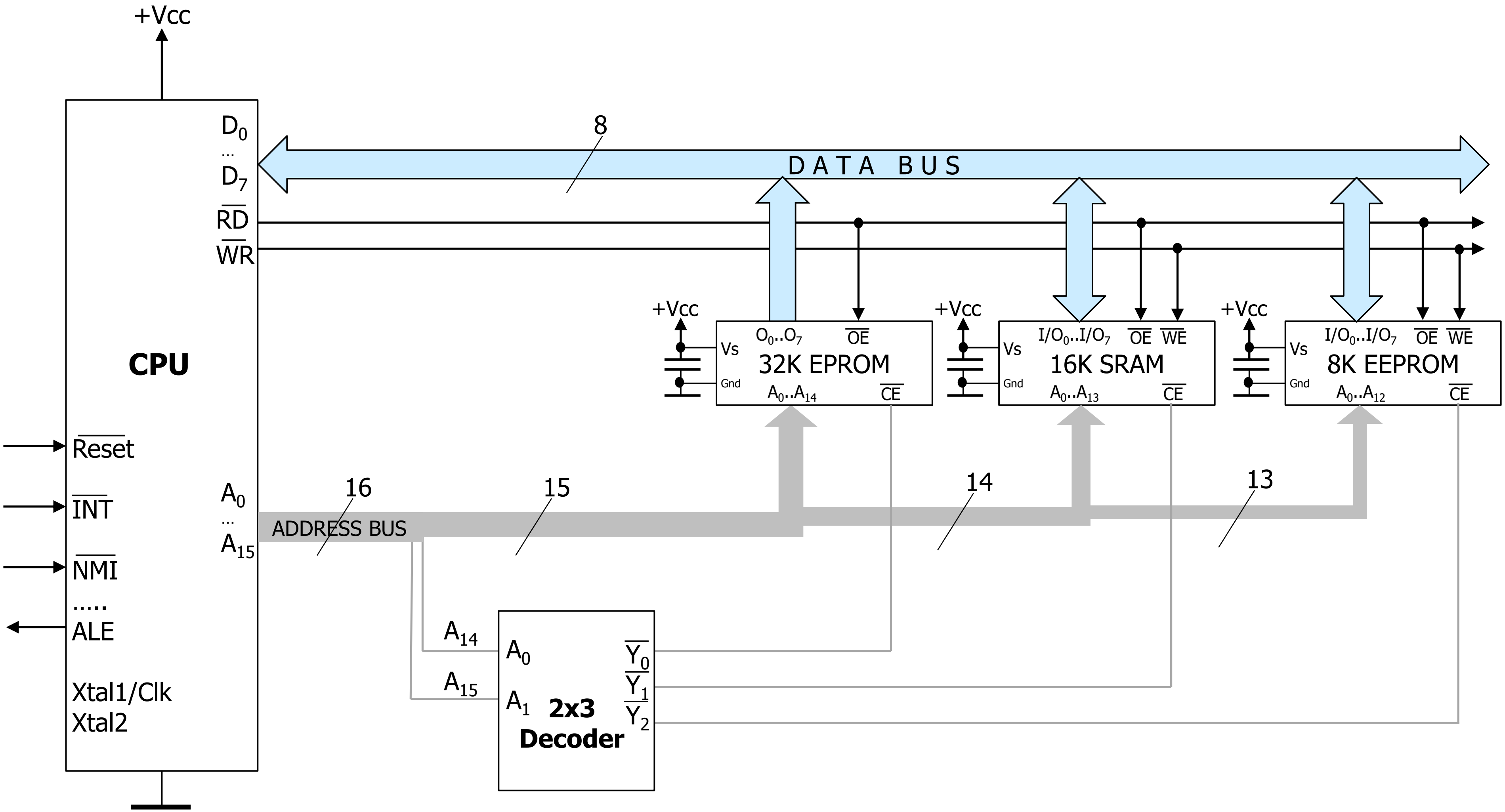
Address lines→ A0..A12





Address Decoding

CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H, 32K EPROM, 16K SRAM, 8K EEPROM



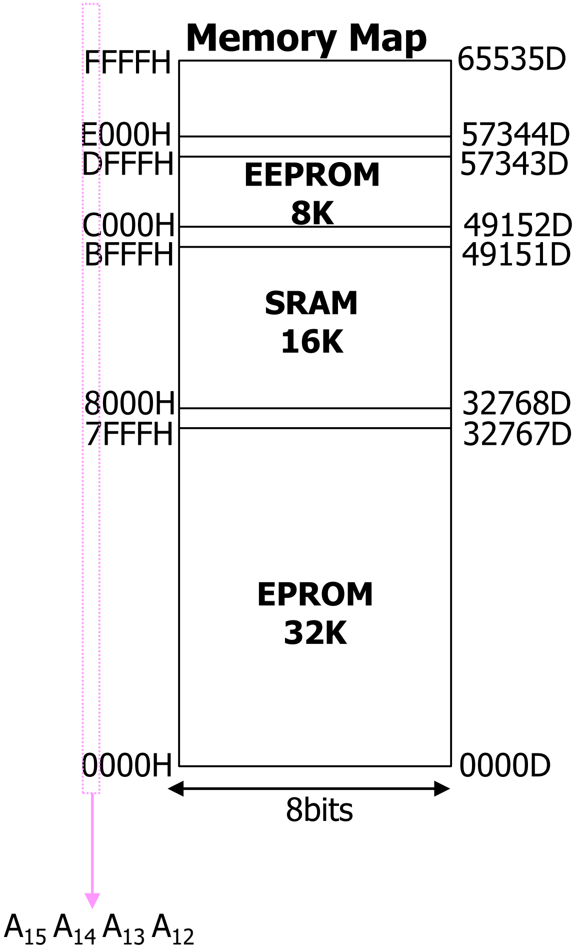
Address Decoder Design

If last segment will not be used:

	CPU:	A15	A14	Output:			
Segment Size:	Decoder:	A1	A0	Y0'	Y1'	Y2'	
16K		0	0	0	1	1	EPROM
16K		0	1	0	1	1	EPROM
16K		1	0	1	0	1	SRAM
16K		1	1	1	1	0	EEPROM

If last segment will be reserved for future:

	CPU:	A15	A14	A13	Output:			FU	
Segment Size:	Decoder:	A2	A1	A0	Y0'	Y1'	Y2'	Y3'	
8K		0	0	0	0	1	1	1	EPROM
8K		0	0	1	0	1	1	1	EPROM
8K		0	1	0	0	1	1	1	EPROM
8K		0	1	1	0	1	1	1	EPROM
8K		1	0	0	1	0	1	1	SRAM
8K		1	0	1	1	0	1	1	SRAM
8K		1	1	0	1	1	0	1	EEPROM
8K		1	1	1	1	1	1	0	Future Use



Memory Map

FFFFH 65535D

E000H 57344D

DFFFH 57343D

EEPROM 8K

C000H 49152D

BFFFH 49151D

SRAM 16K

8000H 32768D

7FFFH 32767D

EPROM 32K

0000H 0000D

8bits

A_{15} A_{14} A_{13} A_{12}

Address Decoder Design

If last segment will not be used:

	CPU:	A15	A14	Output:			
Segment Size:	Decoder:	A1	A0	Y0'	Y1'	Y2'	
16K		0	0	0	1	1	EPROM
16K		0	1	0	1	1	EPROM
16K		1	0	1	0	1	SRAM
16K		1	1	1	1	0	EEPROM

Y0′

$A_1 \backslash A_0$	0	1
0	0	0
1	1	1

Y0′=A1= A15 (CPU)

Y1′

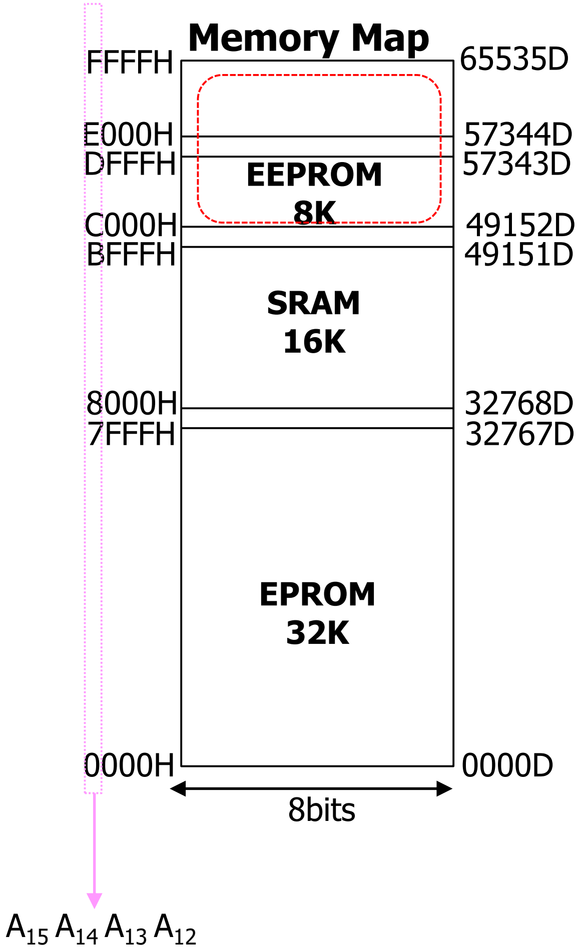
$A_1 \backslash A_0$	0	1
0	1	1
1	0	1

Y1′=(A1A0′)′= A1′+A0=A15′+A14 (CPU)

Y2′

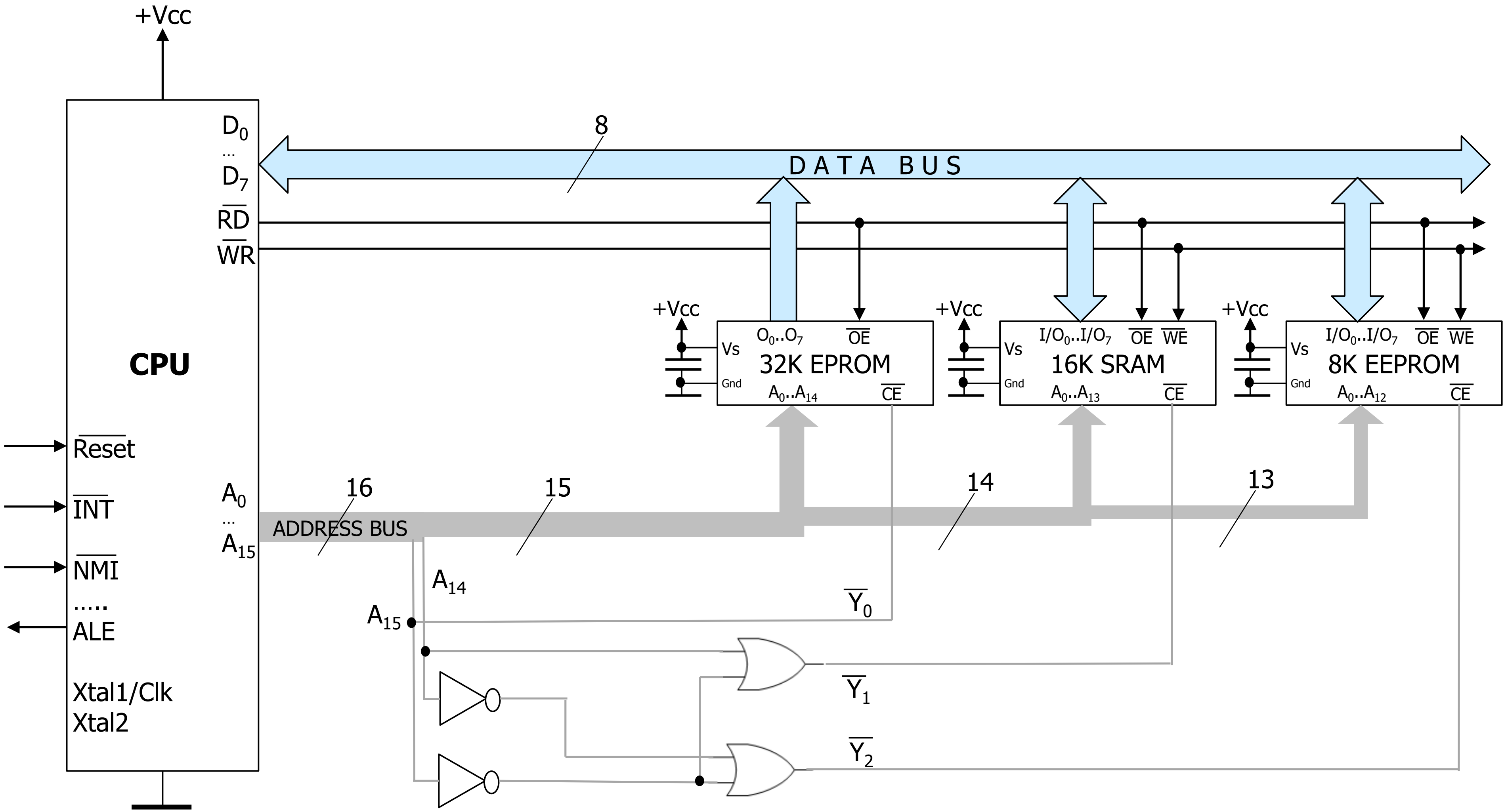
$A_1 \backslash A_0$	0	1
0	1	1
1	1	0

Y2′=(A1A0)′= A1′+A0′=A15′+A14′ (CPU)



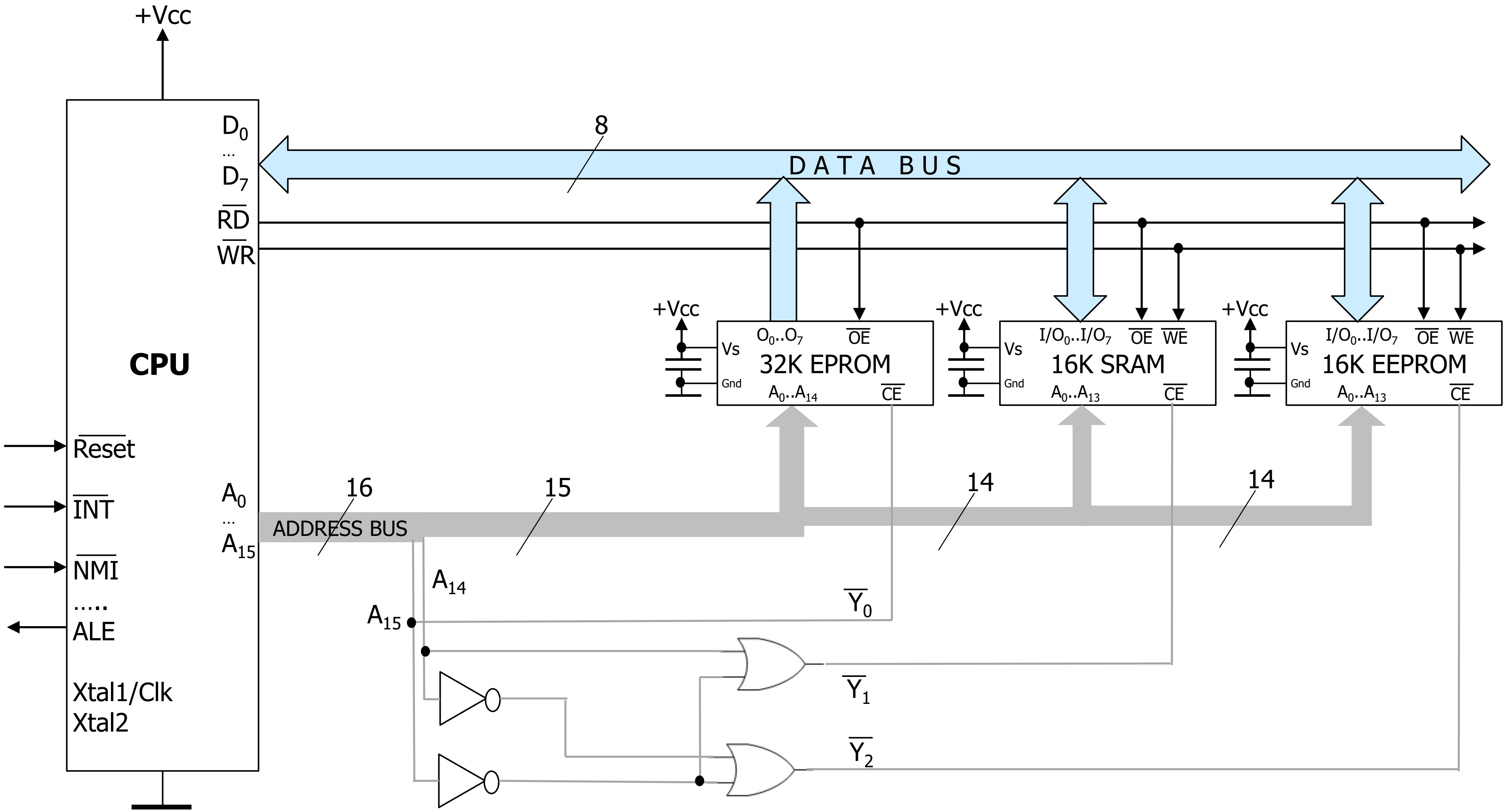
Memory Address Decoding

CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H, 32K EPROM, 16K SRAM, 8K EEPROM



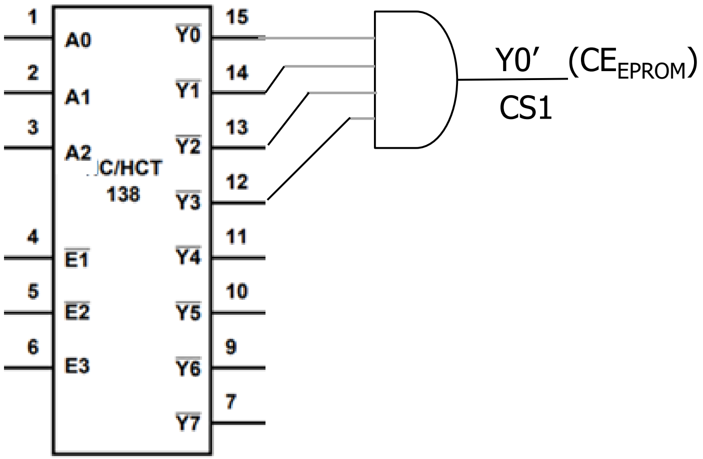
Memory Address Decoding

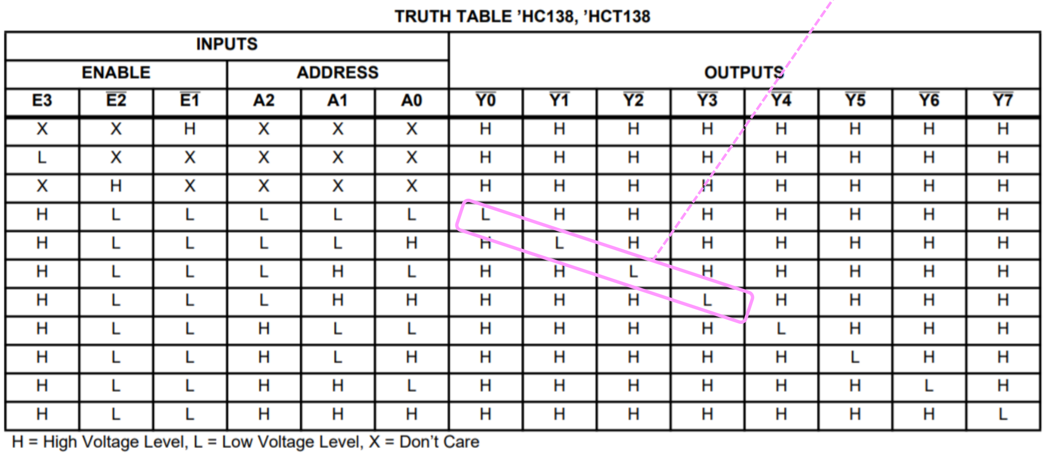
CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H, 32K EPROM, 16K SRAM, 16K EEPROM



If last segment will be reserved for future:

	CPU:	A15	A14	A13	Output:			FU	
Segment Size:	Decoder:	A2	A1	A0	Y0'	Y1'	Y2'	Y3'	
8K		0	0	0	0	1	1	1	EPROM
8K		0	0	1	0	1	1	1	EPROM
8K		0	1	0	0	1	1	1	EPROM
8K		0	1	1	0	1	1	1	EPROM
8K		1	0	0	1	0	1	1	SRAM
8K		1	0	1	1	0	1	1	SRAM
8K		1	1	0	1	1	0	1	EEPROM
8K		1	1	1	1	1	1	0	Future Use



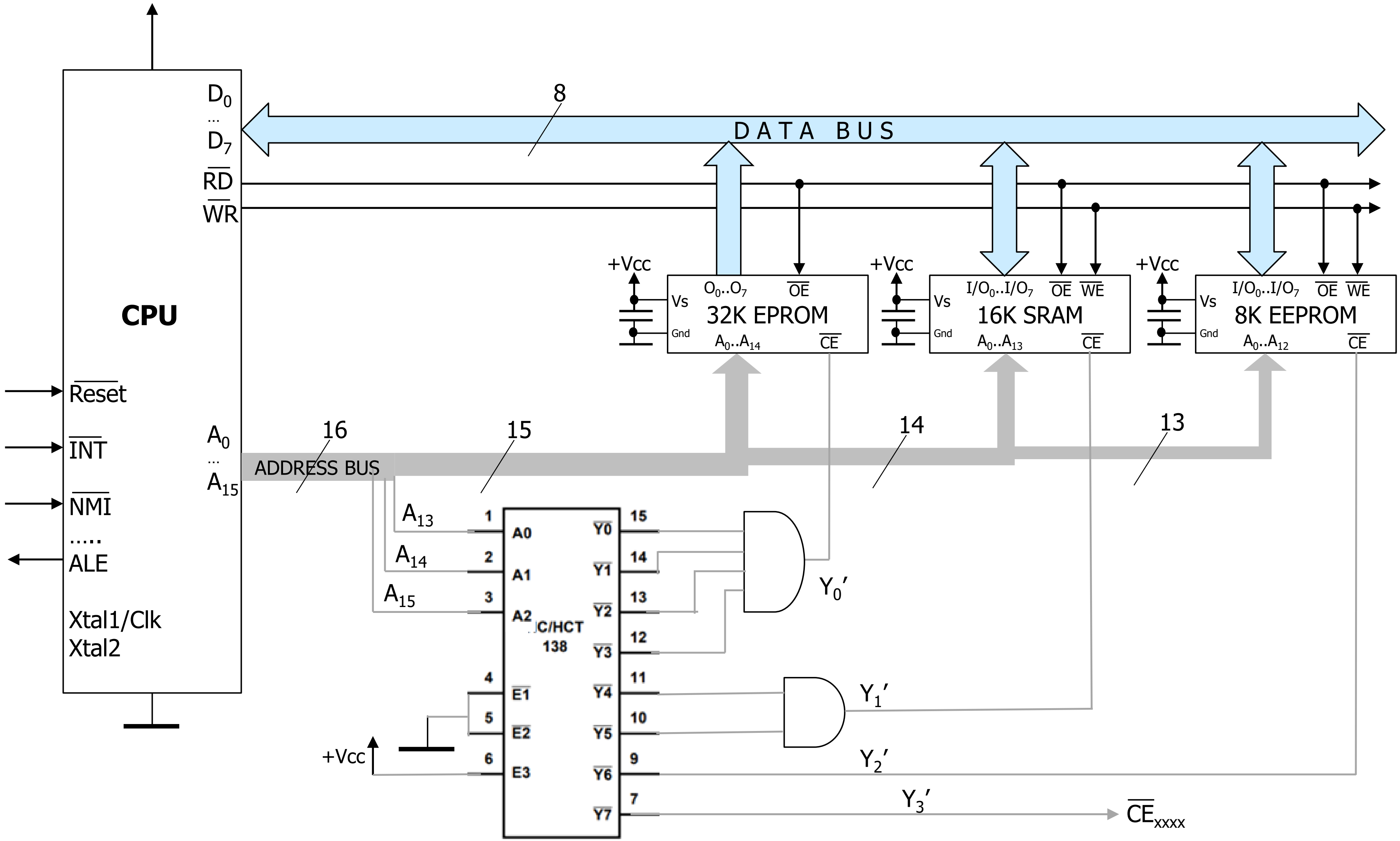

TRUTH TABLE 'HC138, 'HCT138

INPUTS						OUTPUTS							
ENABLE			ADDRESS										
E3	$\overline{E2}$	$\overline{E1}$	A2	A1	A0	$\overline{Y0}$	$\overline{Y1}$	$\overline{Y2}$	$\overline{Y3}$	$\overline{Y4}$	$\overline{Y5}$	$\overline{Y6}$	$\overline{Y7}$
X	X	H	X	X	X	H	H	H	H	H	H	H	H
L	X	X	X	X	X	H	H	H	H	H	H	H	H
X	H	X	X	X	X	H	H	H	H	H	H	H	H
H	L	L	L	L	L	L	H	H	H	H	H	H	H
H	L	L	L	L	H	H	L	H	H	H	H	H	H
H	L	L	L	H	L	H	H	L	H	H	H	H	H
H	L	L	L	H	H	H	H	H	L	H	H	H	H
H	L	L	H	L	L	H	H	H	H	L	H	H	H
H	L	L	H	L	H	H	H	H	H	H	L	H	H
H	L	L	H	H	L	H	H	H	H	H	H	L	H
H	L	L	H	H	H	H	H	H	H	H	H	H	L

H = High Voltage Level, L = Low Voltage Level, X = Don't Care

Memory Address Decoding

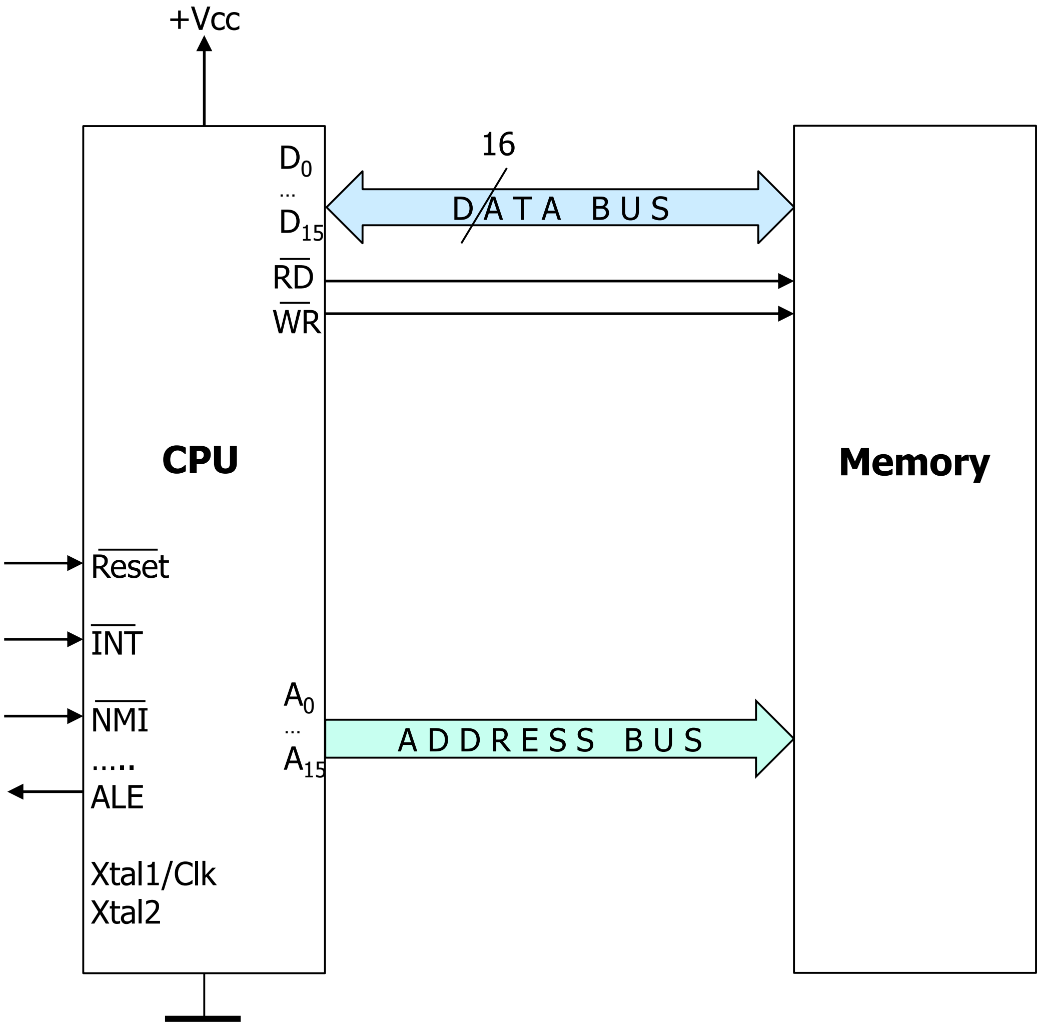
CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H, 32K EPROM, 16K SRAM, 8K EEPROM, 8K xxxx

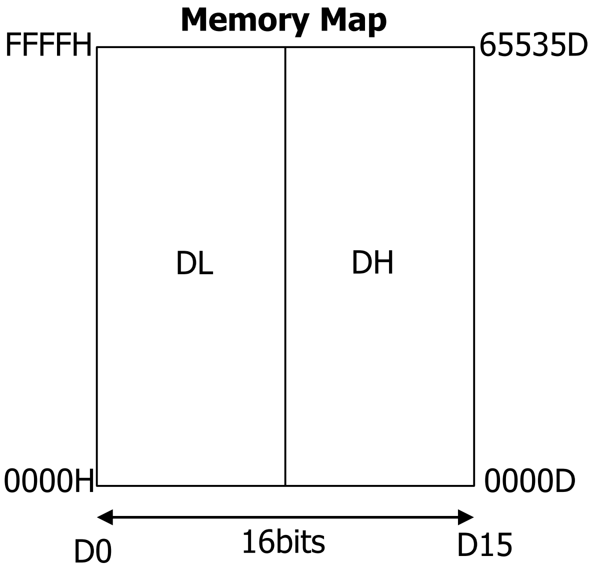


CPU: 64K Addressing Capability, 8Bits, Reset Vector (POR): 0000H

Example 2

CPU: 16bit Addressing Capability, 16Bits, Reset Vector (POR): 0000H, 2x27C128 EPROM, 2x62C256 SRAM 1x28C64 EEPROM





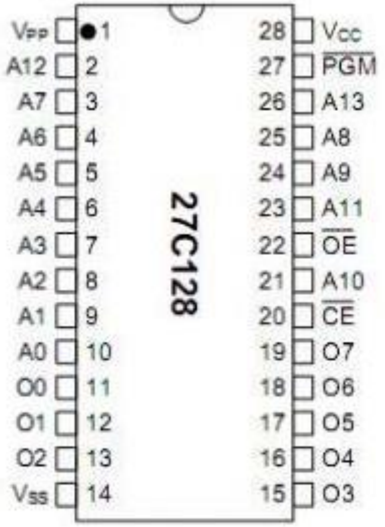
EPROM #1 in memory map

CPU: 16bit Addressing Capability, 16Bits, Reset Vector (POR): 0000H, 2x27C128 EPROM, 2x62C256 SRAM 1x28C64 EEPROM

1- 27C→CMOS EPROM: $\overline{PGM}$, $\overline{OE}$, $\overline{CE}$

2- 27C→$O_0 \ldots O_7$, Generic code: 27C128

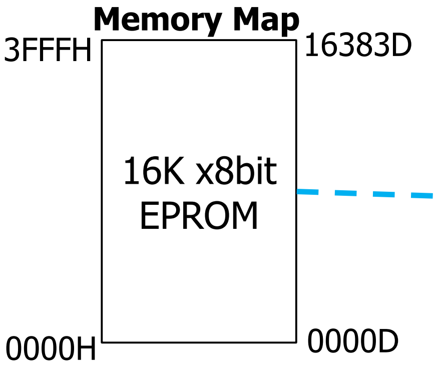
3- x128→ M=128kbit=128x1024 bit

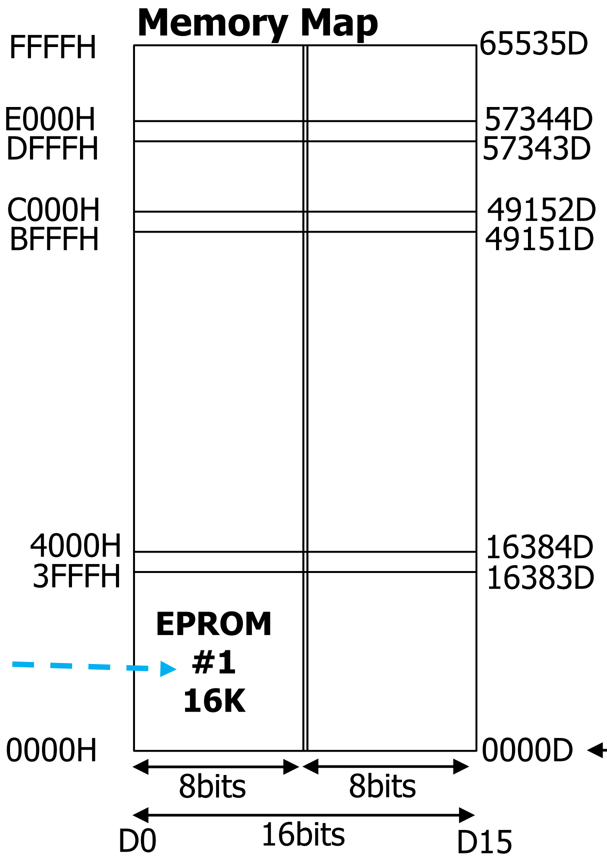


Since 27C refers 8 bits I/O,

Address bus size= $\log_2(128x1024/8)=7+10-3=14$

Address lines→ A0..A13

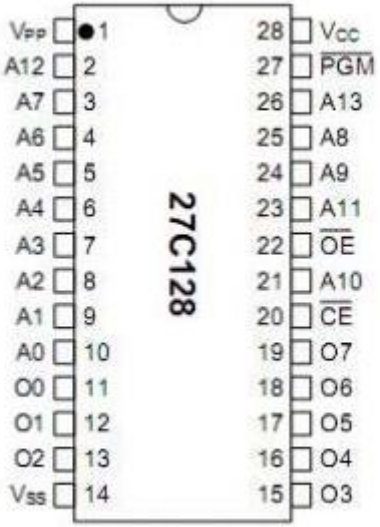




Reset Vector

EPROM #2 in memory map

CPU: 16bit Addressing Capability, 16Bits, Reset Vector (POR): 0000H, 2x27C128 EPROM, 2x62C256 SRAM 1x28C64 EEPROM



1- 27C→CMOS EPROM: $\overline{PGM}$, $\overline{OE}$, $\overline{CE}$

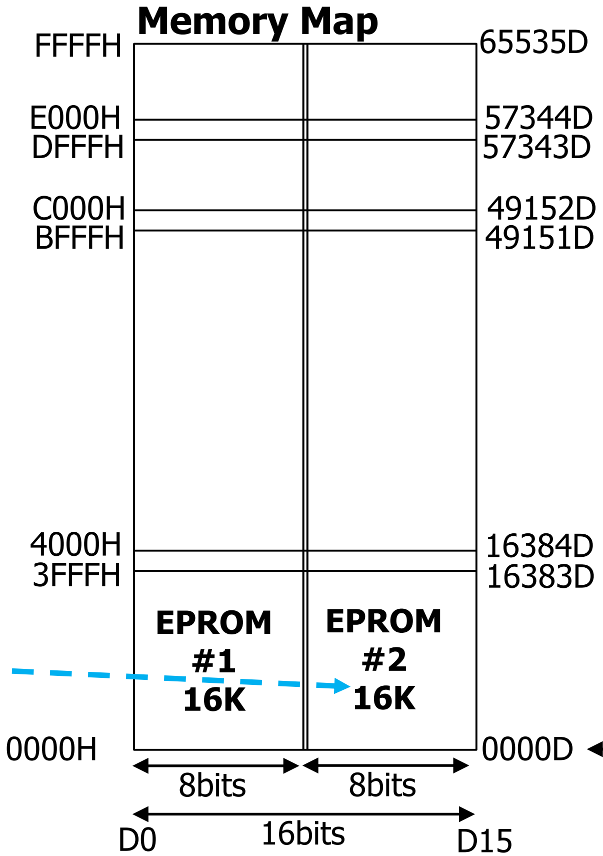
2- 27C→O_0…O_7 , Generic code: 27C128

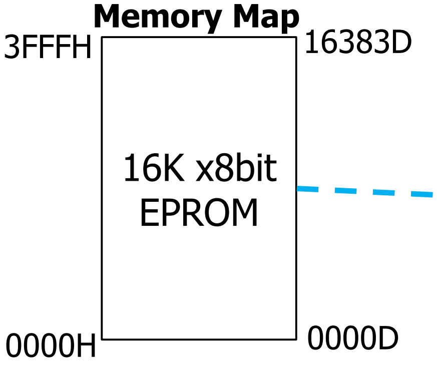
3- x128→ M=128kbit=128x1024 bit

Since 27C refers 8 bits I/O,

Address bus size= $\log_2$ (128x1024/8)=7+10-3=14

Address lines→ A0..A13





Reset Vector

SRAM #1 in memory map

CPU: 16bit Addressing Capability, 16Bits, Reset Vector (POR): 0000H, 2x27C128 EPROM, 2x62C256 SRAM 1x28C64 EEPROM

1- 62C→CMOS SRAM: $\overline{OE}$,$\overline{WE}$,$\overline{CE}$

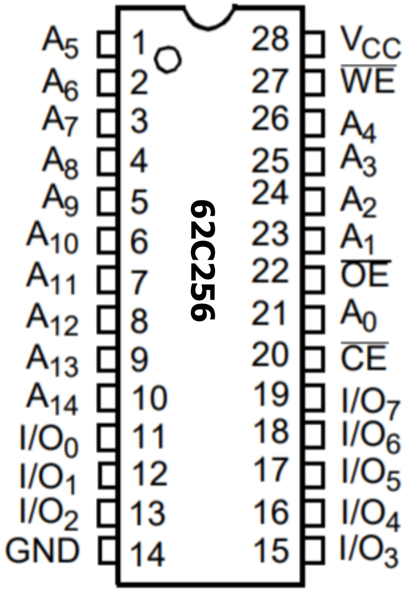
2- 62C→$O_0 \ldots O_7$, Generic code: 62C256

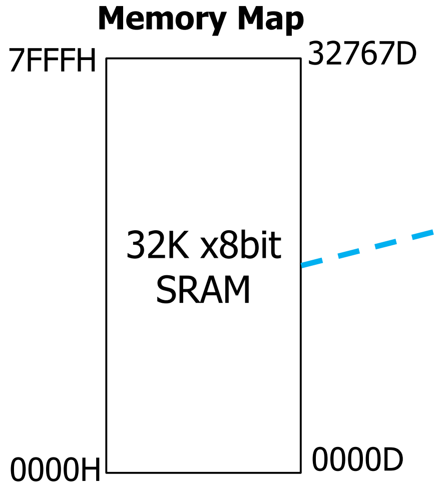
3- x256→ M=256kbit=256x1024 bit

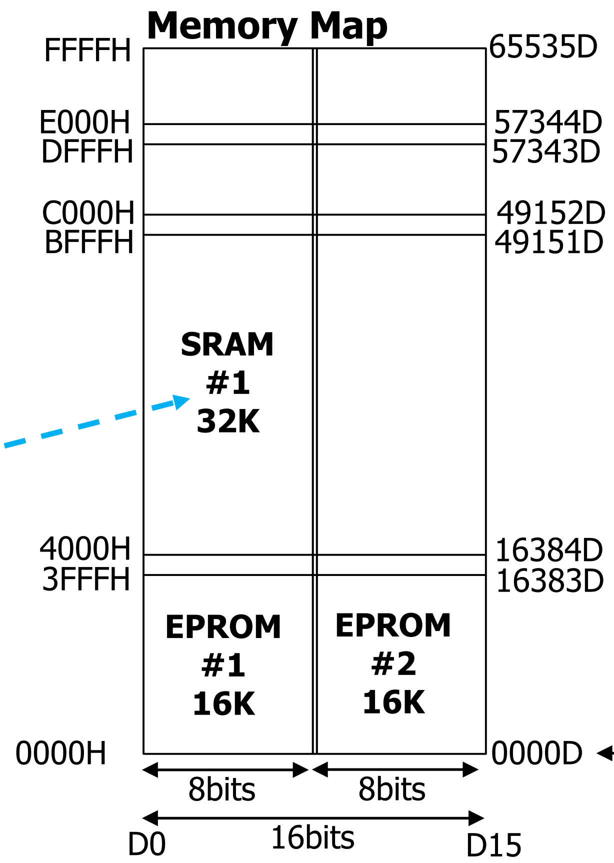
Since 62C refers 8 bits I/O,

Address bus size= $\log_2$ (256x1024/8)=8+10-3=15

Address lines→ A0..A14







Reset Vector

SRAM #2 in memory map

CPU: 16bit Addressing Capability, 16Bits, Reset Vector (POR): 0000H, 2x27C128 EPROM, 2x62C256 SRAM 1x28C64 EEPROM

1- 62C→CMOS SRAM: $\overline{OE}$,$\overline{WE}$,$\overline{CE}$

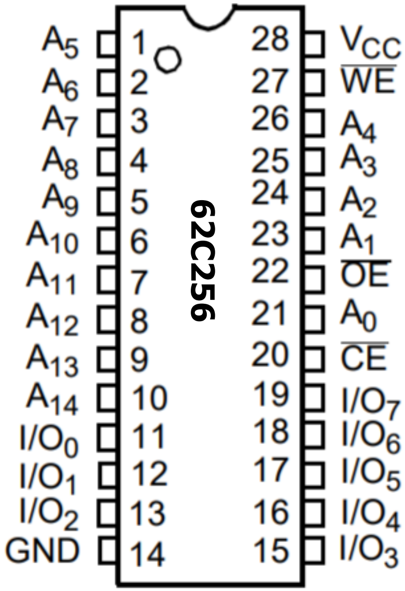
2- 62C→$O_0 \ldots O_7$, Generic code: 62C256

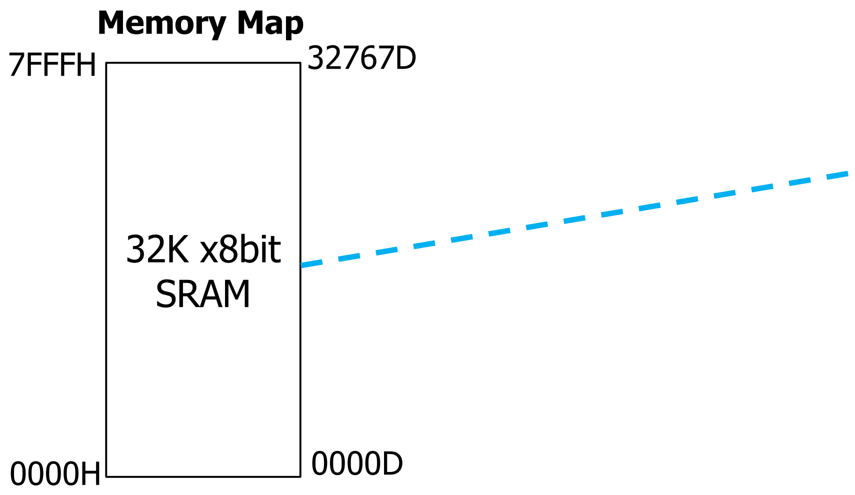
3- x256→ M=256kbit=256x1024 bit

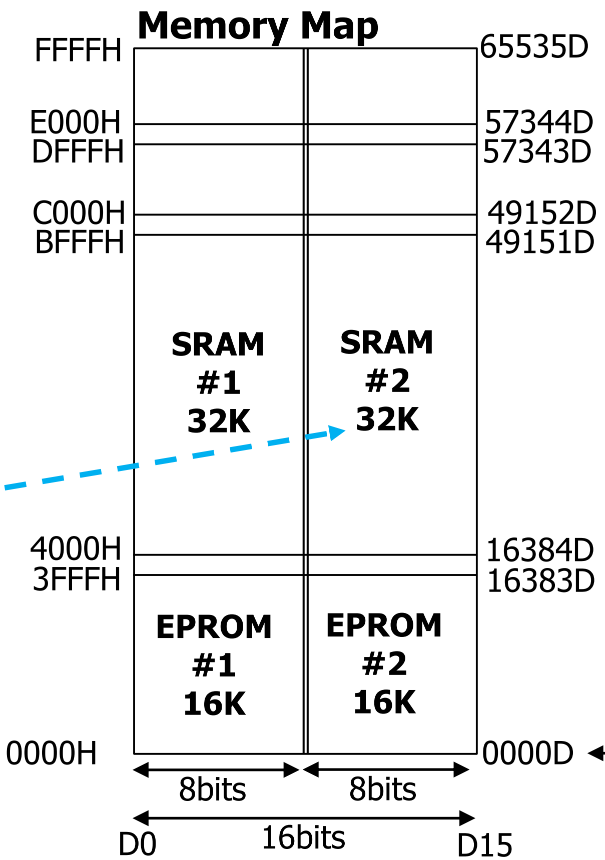
Since 62C refers 8 bits I/O,

Address bus size= $\log_2$ (256x1024/8)=8+10-3=15

Address lines→ A0..A14







Reset Vector

EEPROM in memory map

CPU: 16bit Addressing Capability, 16Bits, Reset Vector (POR): 0000H, 2x27C128 EPROM, 2x62C256 SRAM 1x28C64 EEPROM

1- 28C→CMOS EPROM: $\overline{OE}$, $\overline{WE}$, $\overline{CE}$

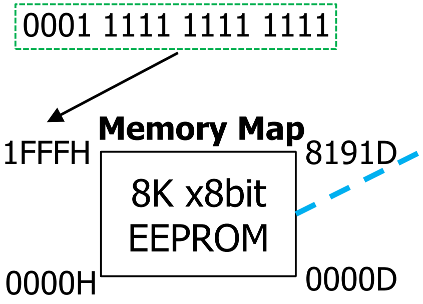
2- 28C→$O_0 \ldots O_7$, Generic code: 28C64

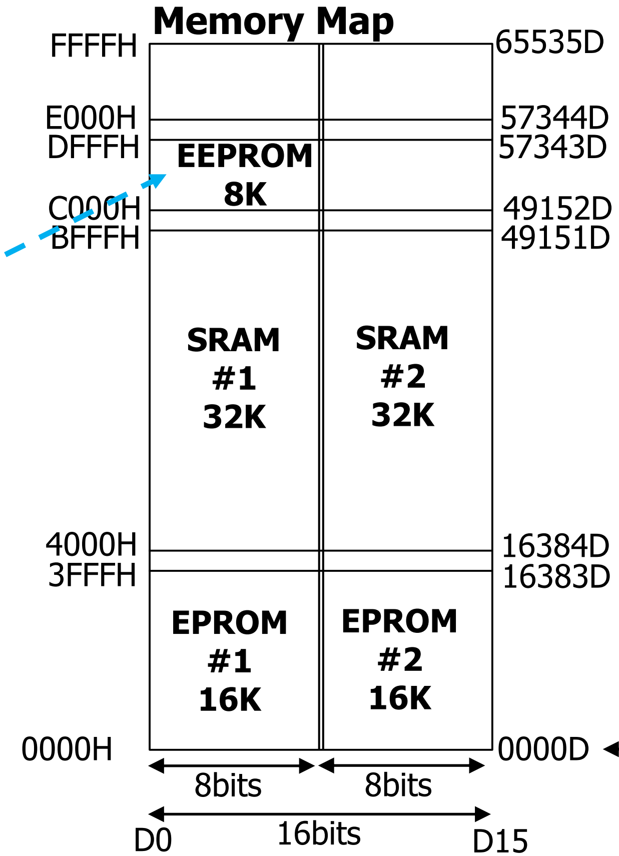
3- x64→ M=64kbit=64x1024 bit

Since 28C refers 8 bits I/O,

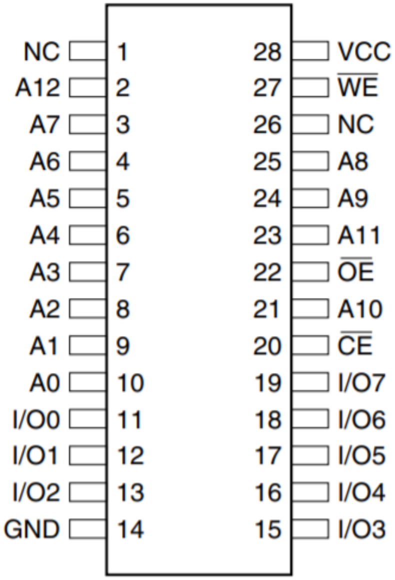
Address bus size= $\log_2$ (64x1024/8)=6+10-3=13

Address lines→ A0..A12





Reset Vector



Address Decoder Design

If last segment will not be used:

Decoder output lines=Amount of segments in the memory map=3

Decoder input lines=log$_2$(CPU Address range/minimum segment size)
=log$_2$(64K-line/16K-line)
=log$_2$(4)=2 → A15, A14

	CPU:	A15	A14	Output:			
Segment Size:	Decoder:	A1	A0	Y0'	Y1'	Y2'	
16K		0	0	0	1	1	EPROM#1,#2
16K		0	1	1	0	1	SRAM#1,#2
16K		1	0	1	0	1	SRAM#1,#2
16K		1	1	1	1	0	EEPROM

Y0′

A_1 \ A_0	0	1
0	0	1
1	1	1

Y0′=A1+A0= A15+A14 (CPU)

Y1′

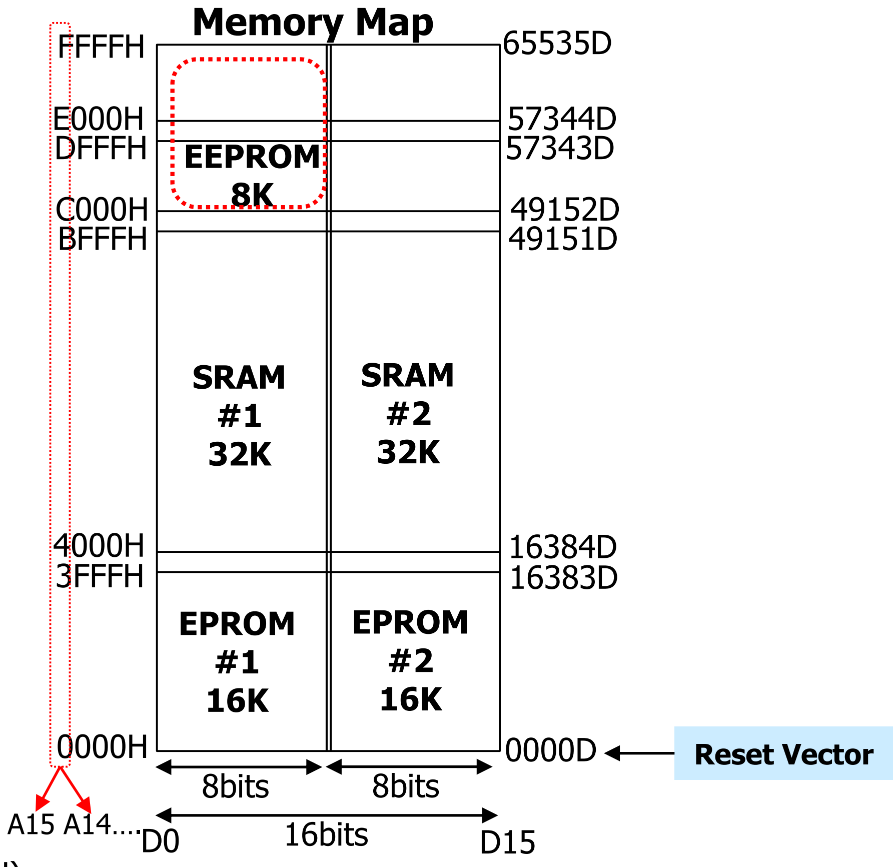
A_1 \ A_0	0	1
0	1	0
1	0	1

Y1′=A0′A1′+A1A0=A14′A15′+A14A15 (CPU)

Y2′

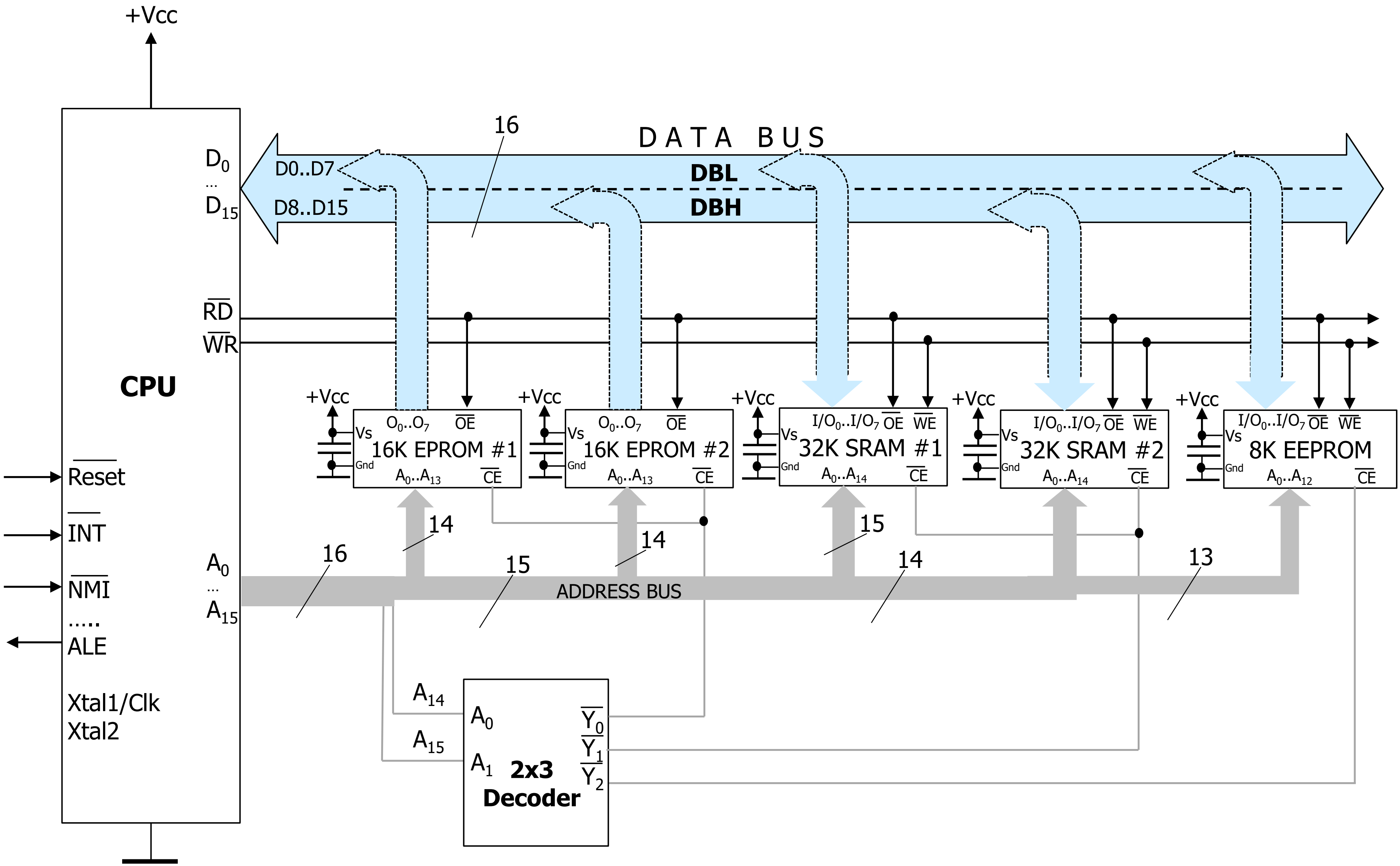
A_1 \ A_0	0	1
0	1	1
1	1	0

Y2′=(A1A0)′= A1′+A0′=A15′+A14′ (CPU)



Example 2 (16bits Data Bus)

CPU: 64K Addressing Capability, 16Bits, Reset Vector (POR): 0000H, 2x27C128 EPROM, 2x62C256 SRAM 1x28C64 EEPROM



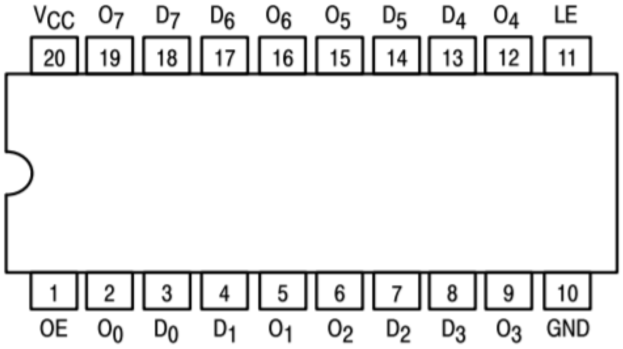
74HC373 3-state D-type latch

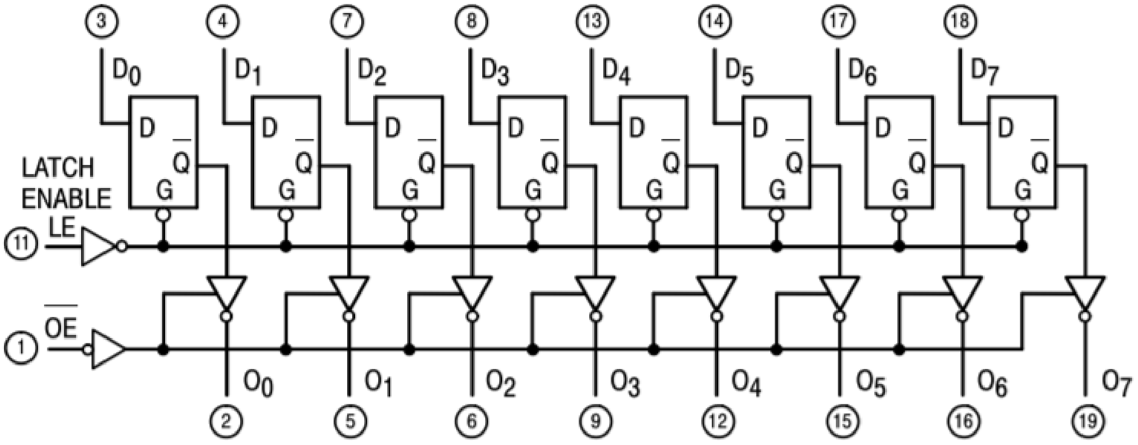
LS373

D_n	LE	$\overline{\text{OE}}$	O_n
H	H	L	H
L	H	L	L
X	L	L	Q_0
X	X	H	Z*

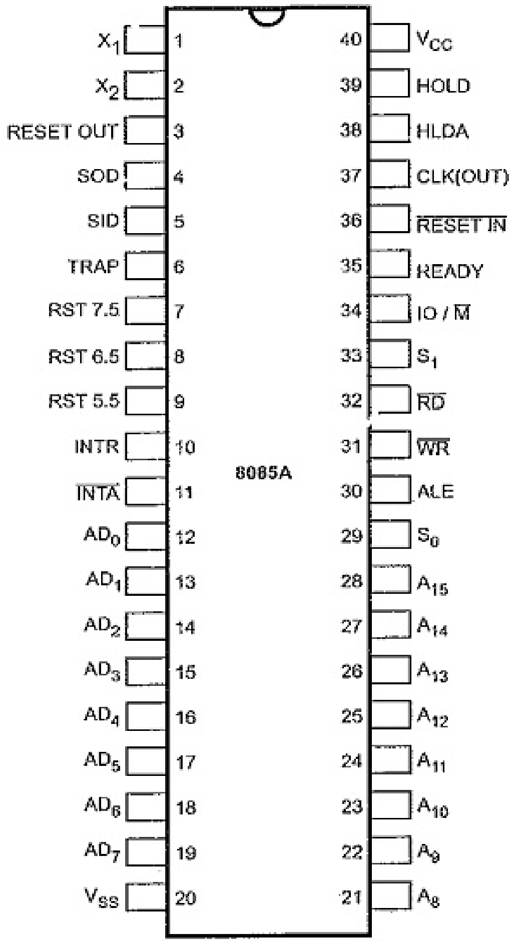
H = HIGH Voltage Level
L = LOW Voltage Level
X = Immaterial
Z = High Impedance

SN54/74LS373

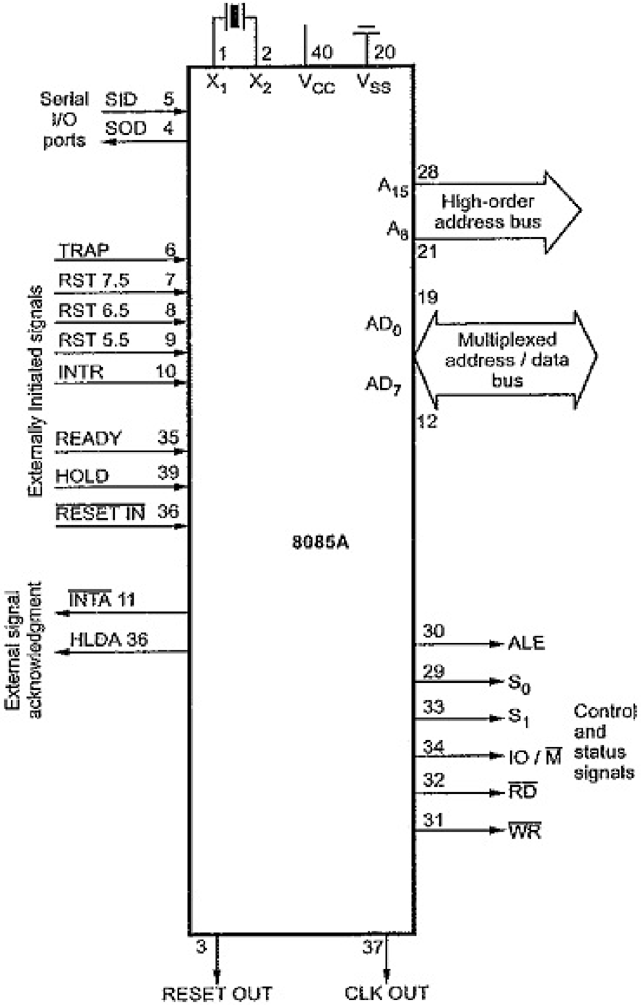




Intel 8085 CPU



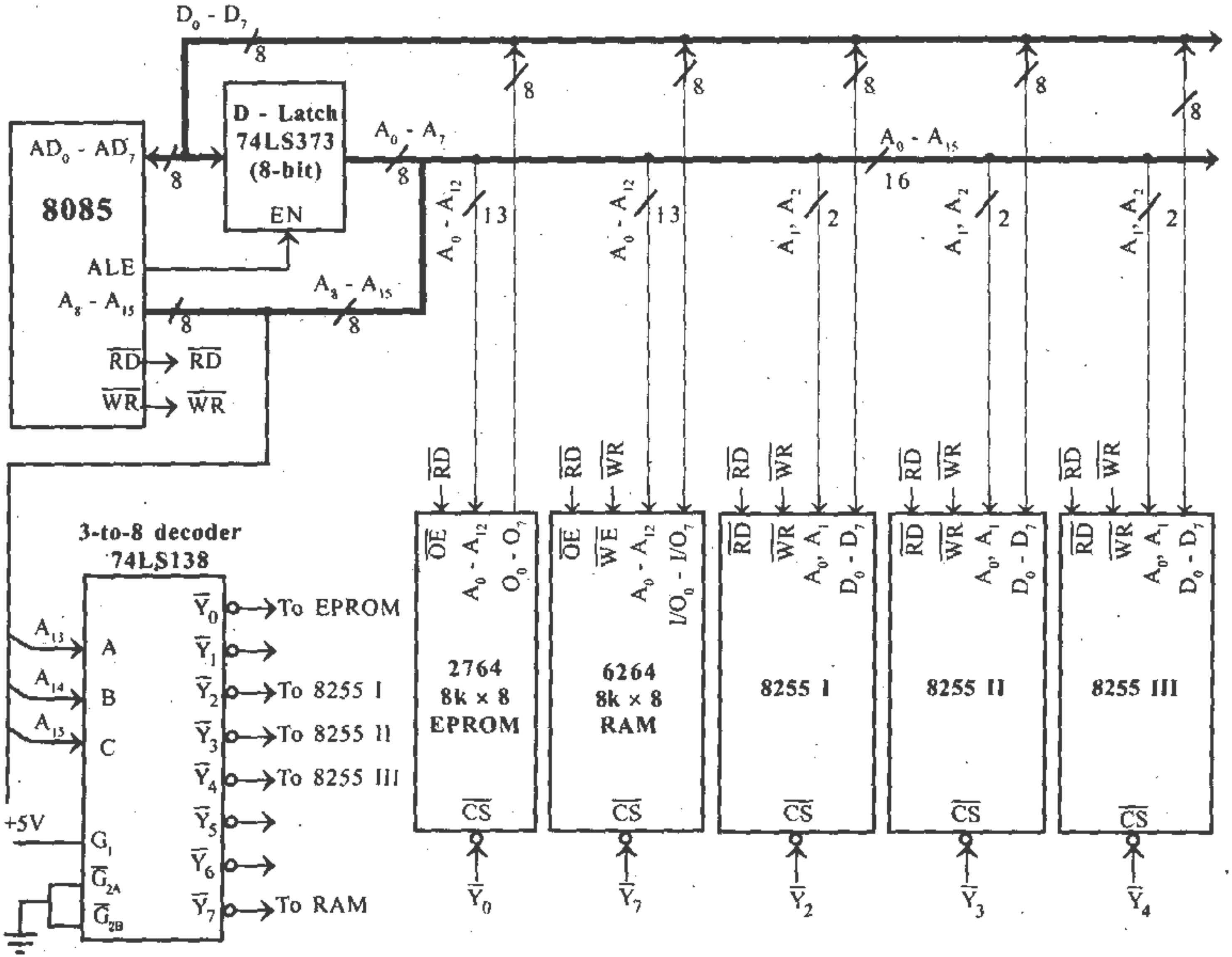
(a) Pin configuration



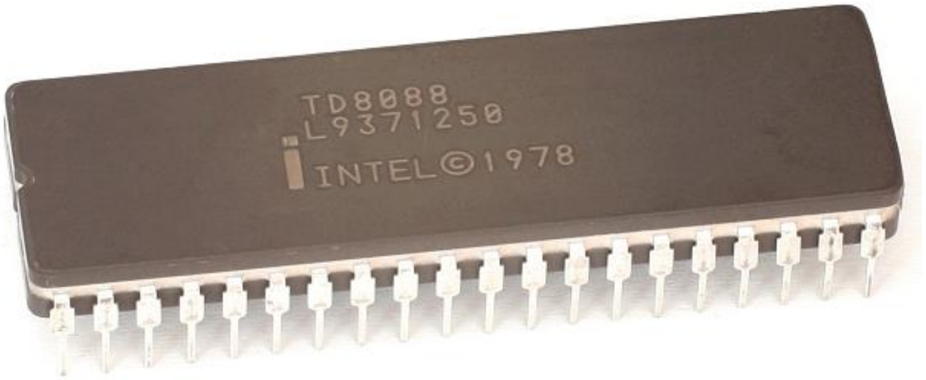
(b) Functional pin diagram

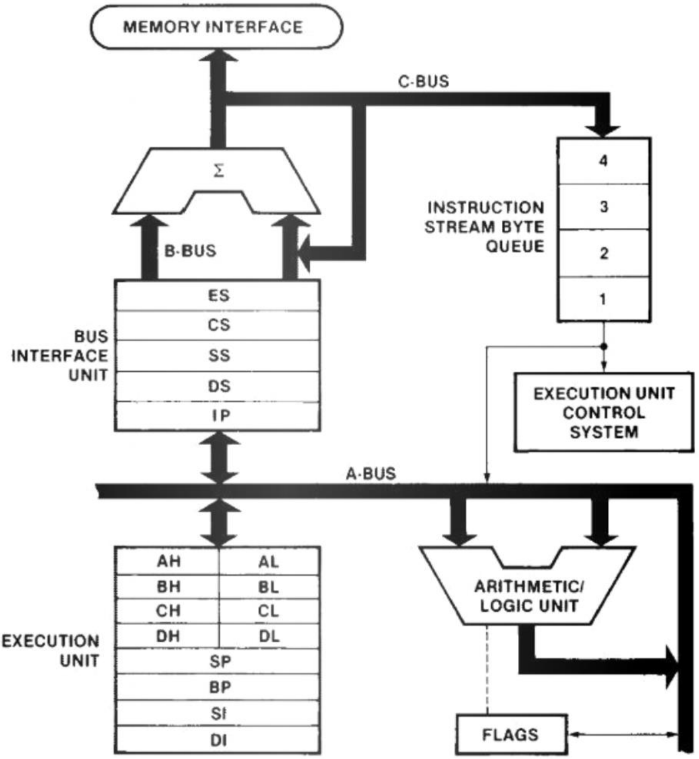
Performance	
Max. CPU clock rate	3, 5 and 6 MHz
Data width	8 Bit
Address width	16 Bit
Architecture and classification	
Min. feature size	3 µm
Instruction set	8085
Physical specifications	
Transistors	•6,500
Package(s)	•40-pin DIP
Socket(s)	•DIP40

CPU: 8085 27C64 EPROM, 62C64 SRAM 3x8255 I/O Ports

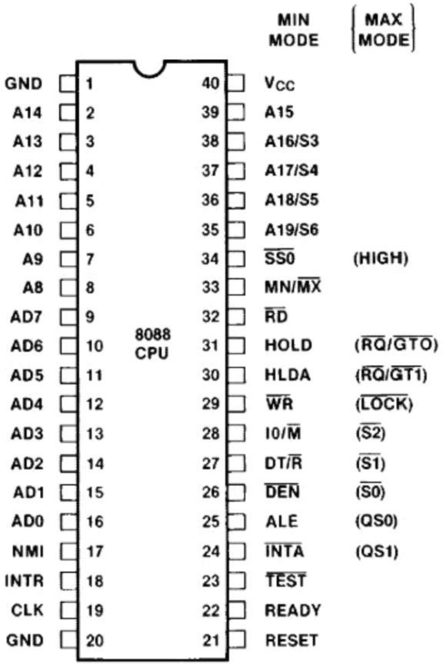


8088 CPU





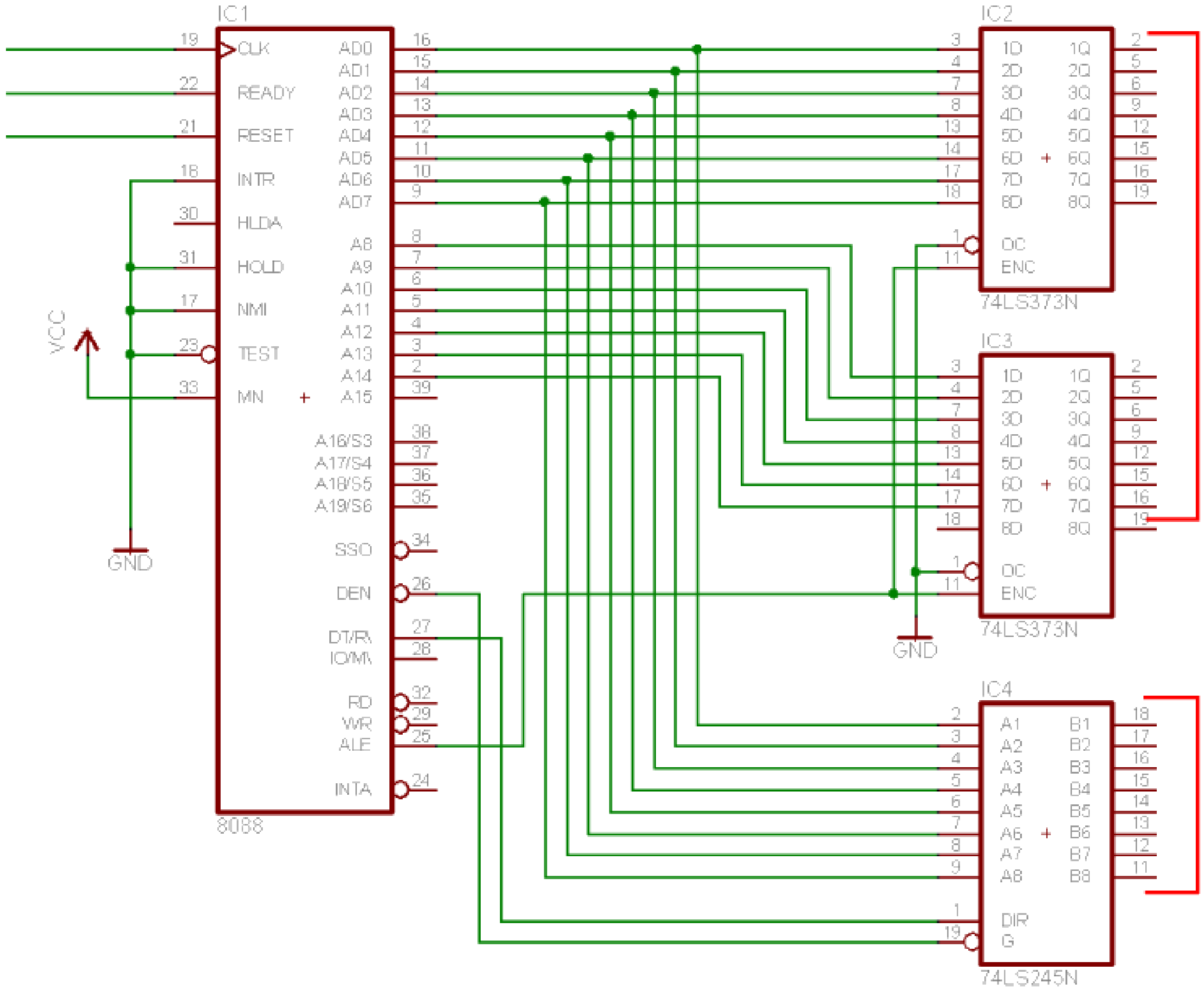
8088 CPU Functional Block Diagram



8088 Pin Configuration

Max. CPU clock rate	5 MHz to 16 MHz
Data width	8 bits
Address width	20 bits
Architecture and classification	
Min. feature size	3 µm
Instruction set	x86-16
Physical specifications	
Transistors	•29,000
Co-processor	Intel 8087
Package(s)	•40-pin DIP •44-pin PLCC

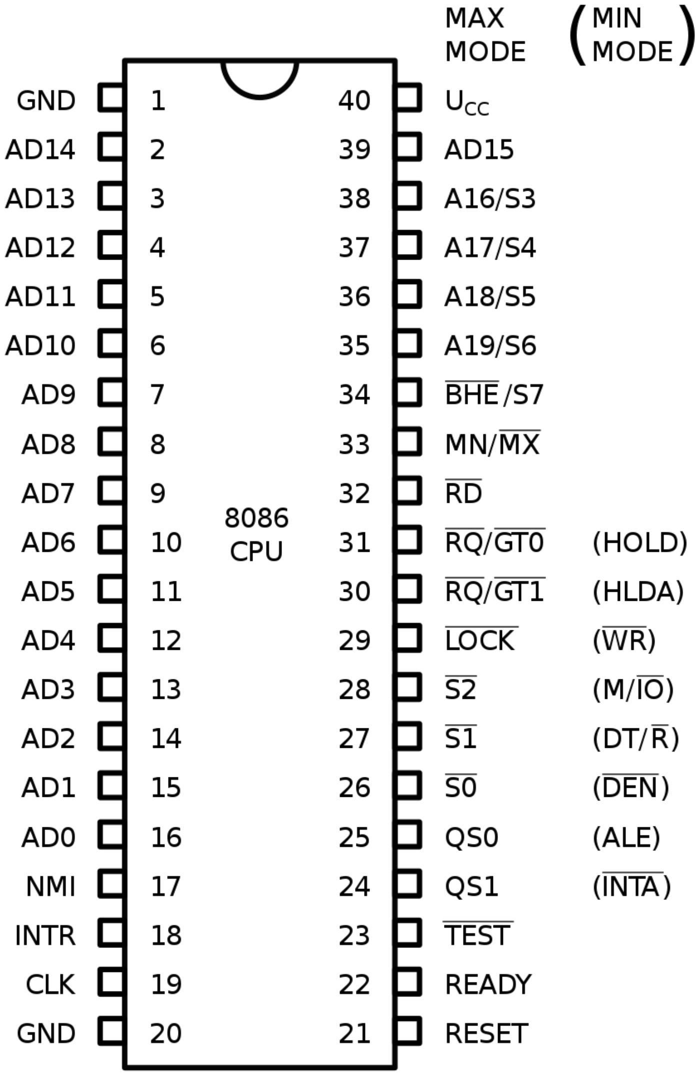
8088 Memory Interfacing

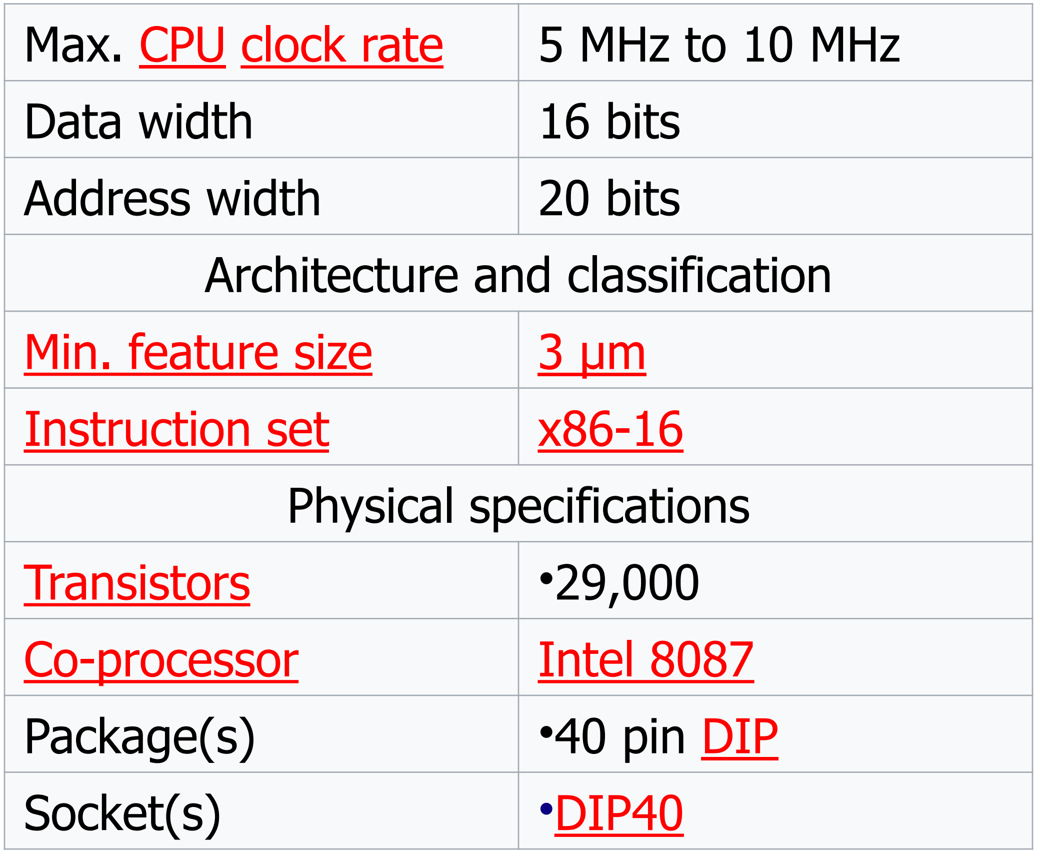


Address bus
(A0-A14)

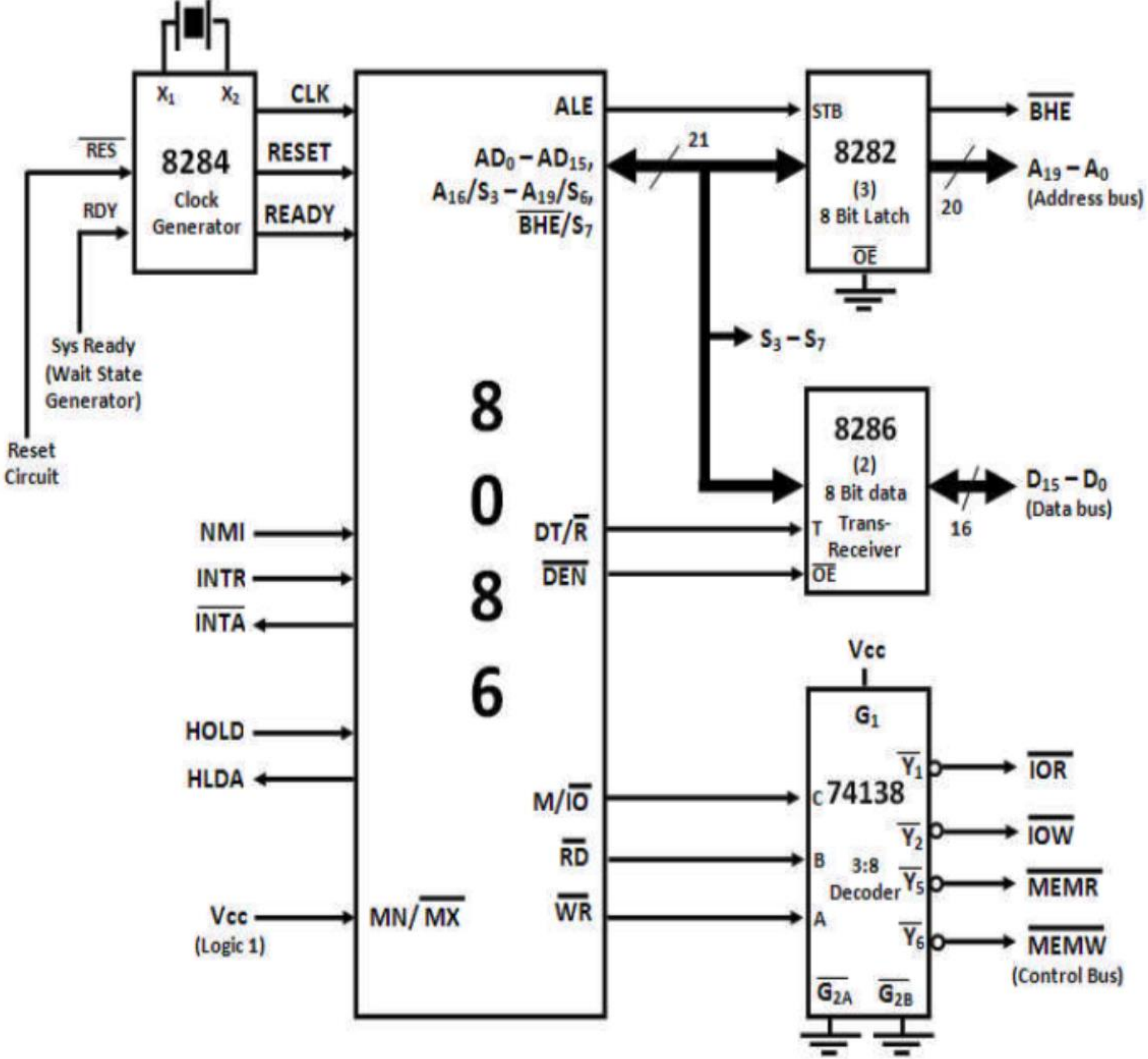
Data bus
(D0-D7)

16-bit processor example (1978-) :

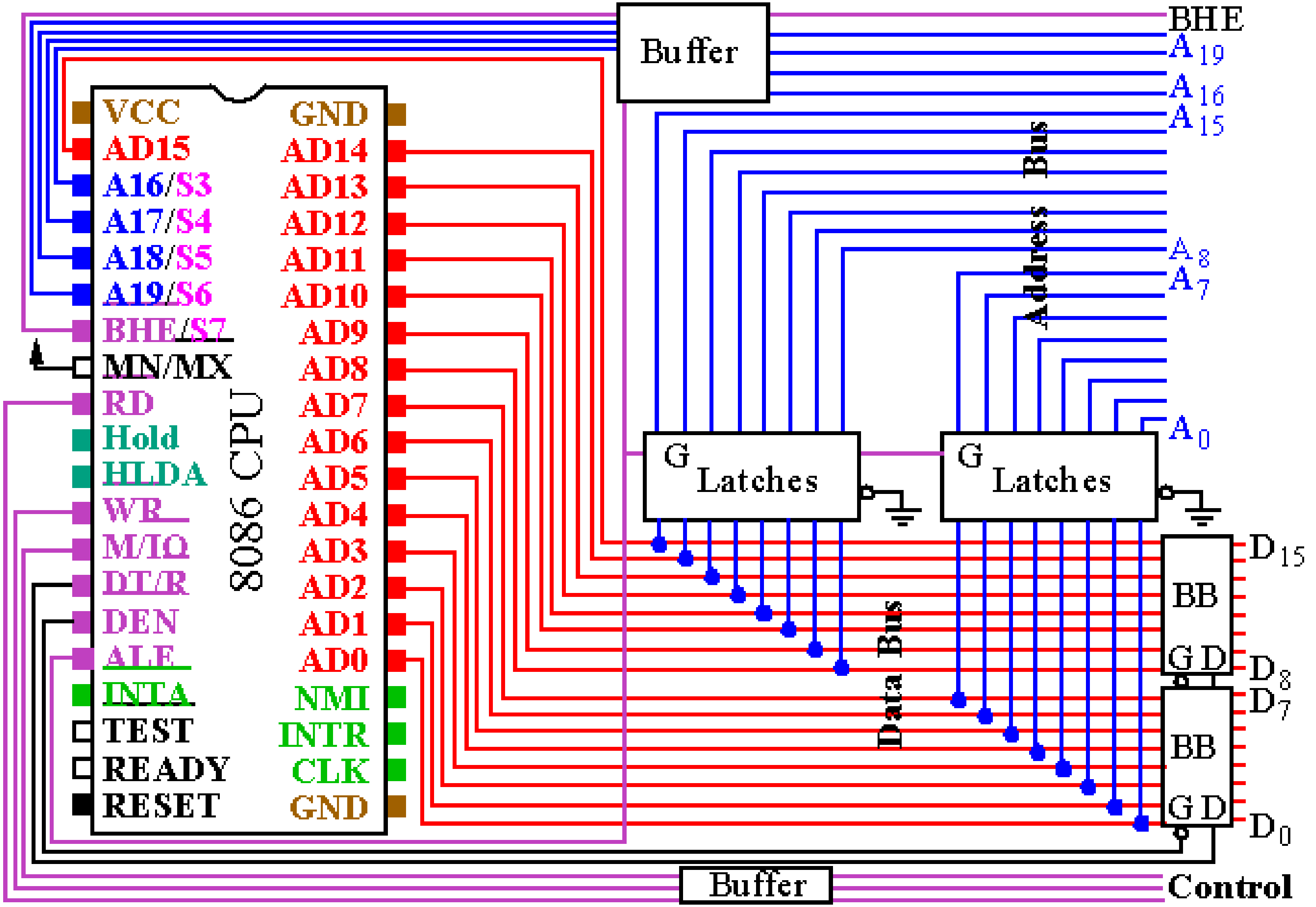




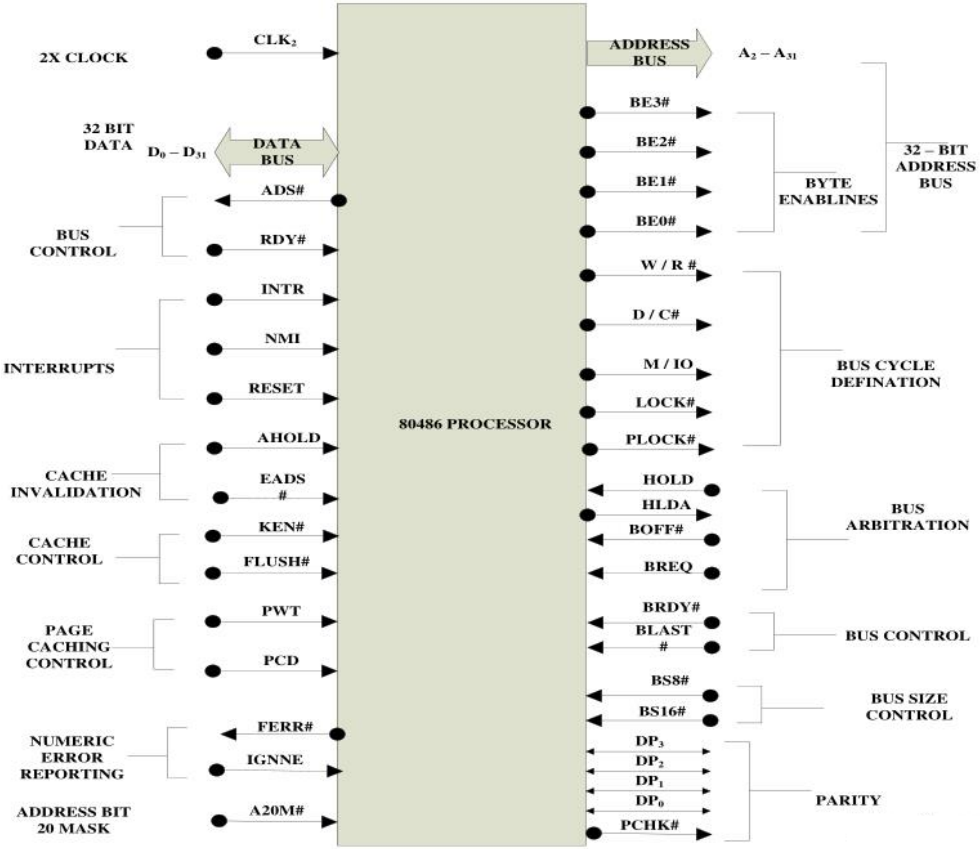
Max. CPU clock rate	5 MHz to 10 MHz
Data width	16 bits
Address width	20 bits
Architecture and classification	
Min. feature size	3 µm
Instruction set	x86-16
Physical specifications	
Transistors	•29,000
Co-processor	Intel 8087
Package(s)	•40 pin DIP
Socket(s)	•DIP40


X_1
X_2
8284
Clock
Generator
$\overline{RES}$
RDY
CLK
RESET
READY
Sys Ready
(Wait State
Generator)
Reset
Circuit
8
0
8
6
ALE
$AD_0 - AD_{15}$,
$A_{16}/S_3 - A_{19}/S_6$,
$\overline{BHE}/S_7$
21
STB
8282
(3)
8 Bit Latch
$\overline{OE}$
$\overline{BHE}$
20
$A_{19} - A_0$
(Address bus)
$S_3 - S_7$
8286
(2)
8 Bit data
Trans-
Receiver
T
$\overline{OE}$
16
$D_{15} - D_0$
(Data bus)
DT/$\overline{R}$
$\overline{DEN}$
NMI
INTR
$\overline{INTA}$
HOLD
HLDA
Vcc
(Logic 1)
MN/$\overline{MX}$
M/$\overline{IO}$
$\overline{RD}$
$\overline{WR}$
Vcc
G_1
74138
3:8
Decoder
C
B
A
$\overline{Y_1}$
$\overline{Y_2}$
$\overline{Y_5}$
$\overline{Y_6}$
$\overline{G_{2A}}$
$\overline{G_{2B}}$
$\overline{IOR}$
$\overline{IOW}$
$\overline{MEMR}$
$\overline{MEMW}$
(Control Bus)

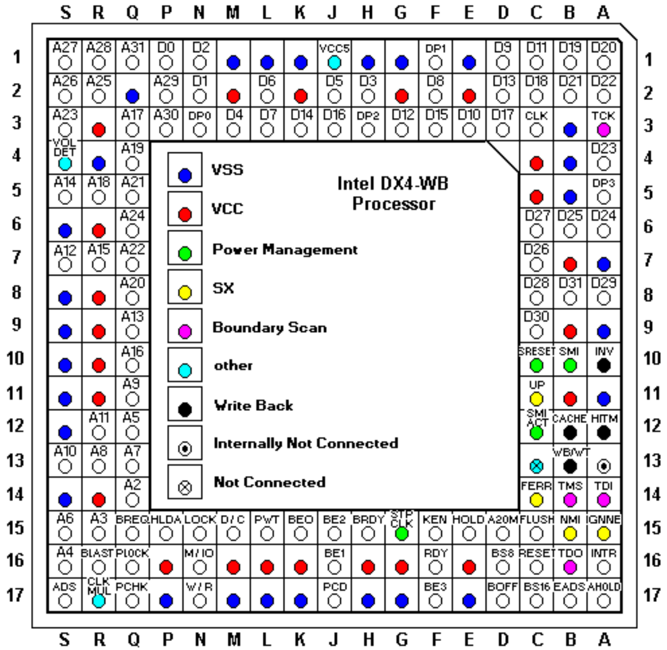
8086 Address and Data Bus Connection in Minimum Mode



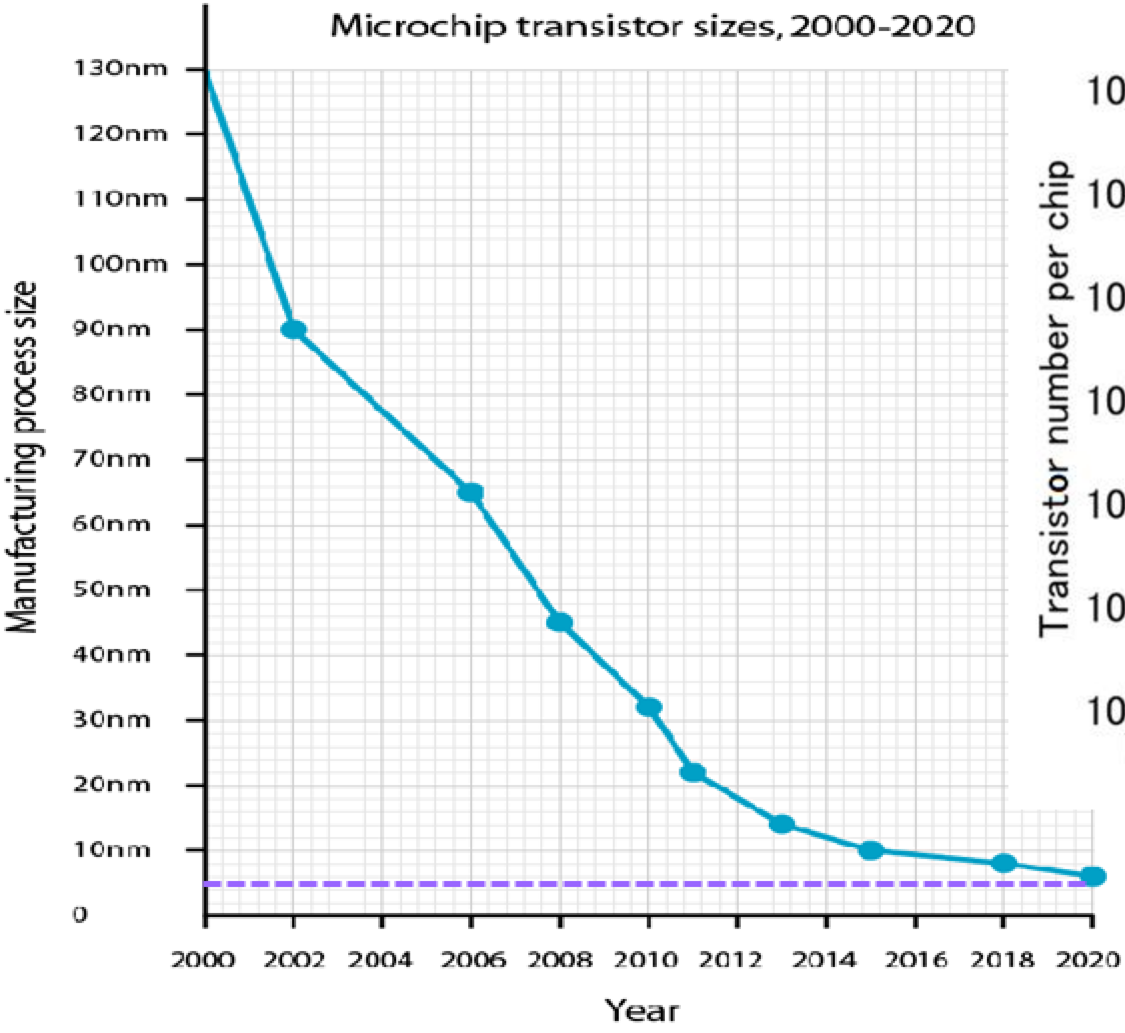
80486 CPU

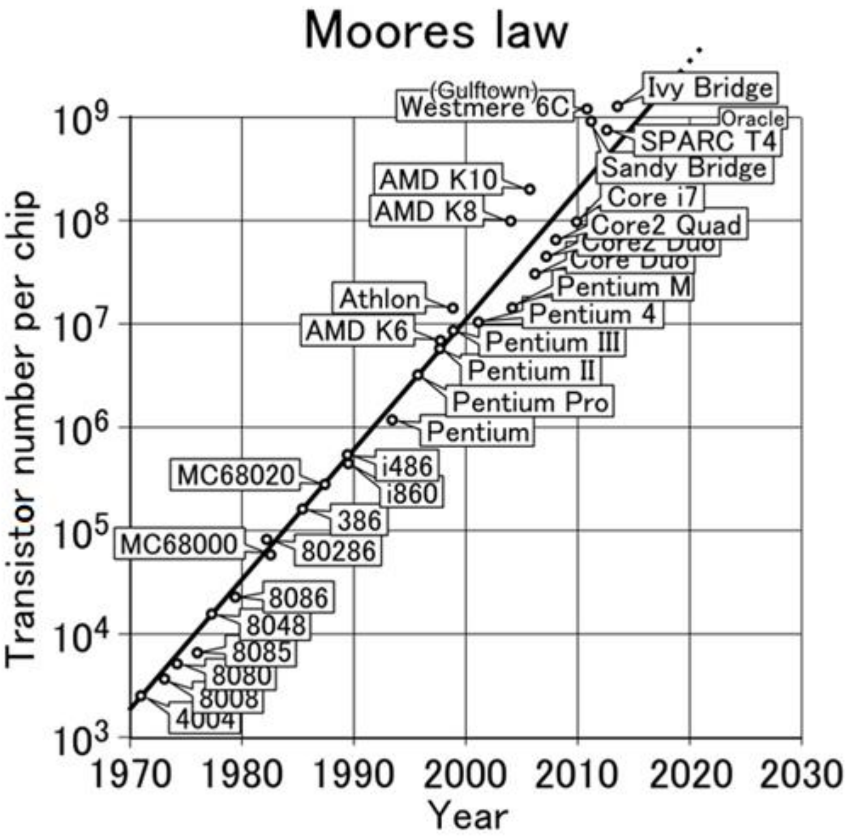


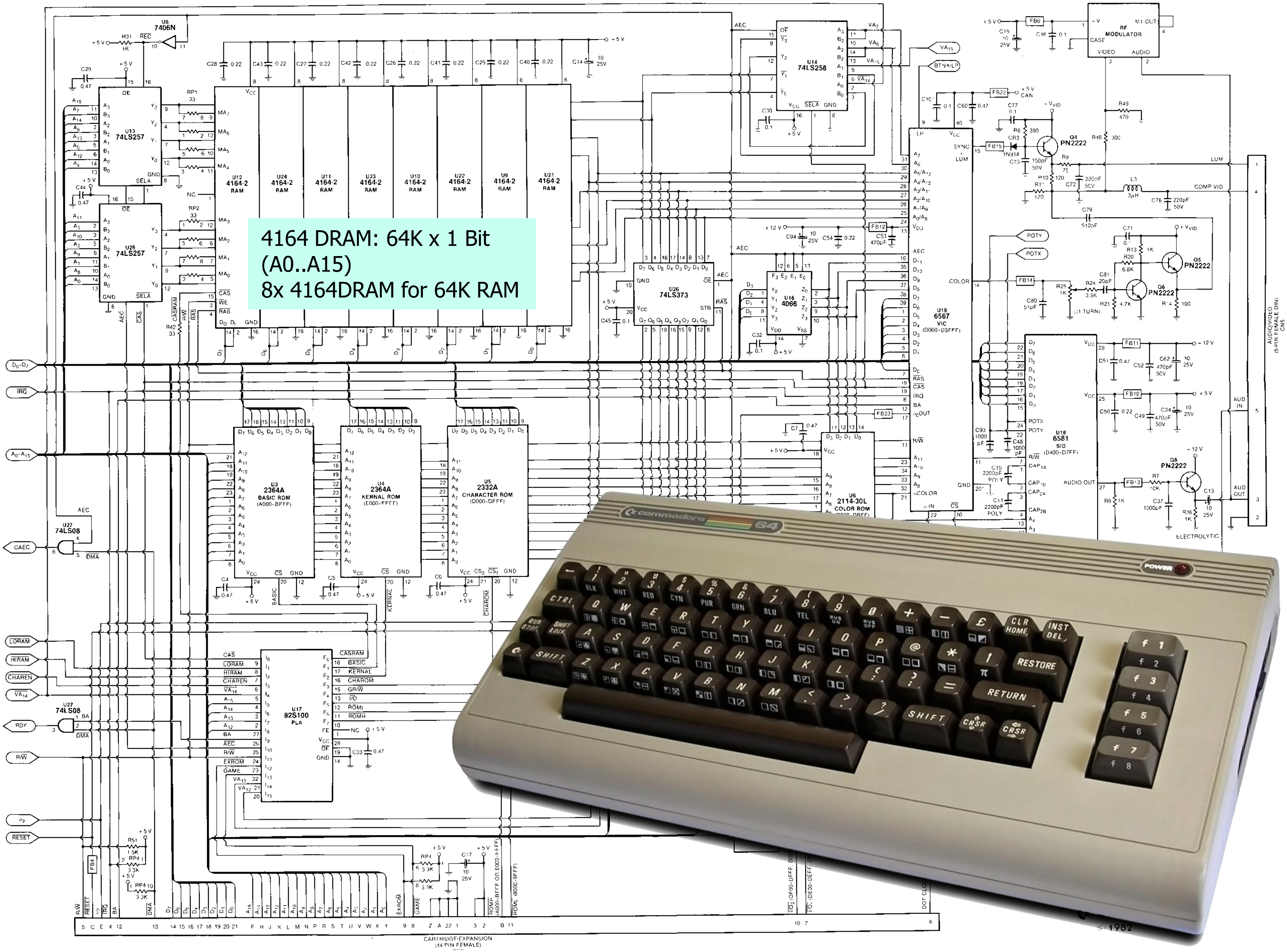
Max. CPU clock rate	16 MHz to 100 MHz
FSB speeds	16 MHz to 50 MHz
Data width	32 bits
Address width	32 bits
Virtual address width	32 bits (linear); 46 bits (logical)
Architecture and classification	
Min. feature size	1 µm to 0.6 µm



Historical Developement



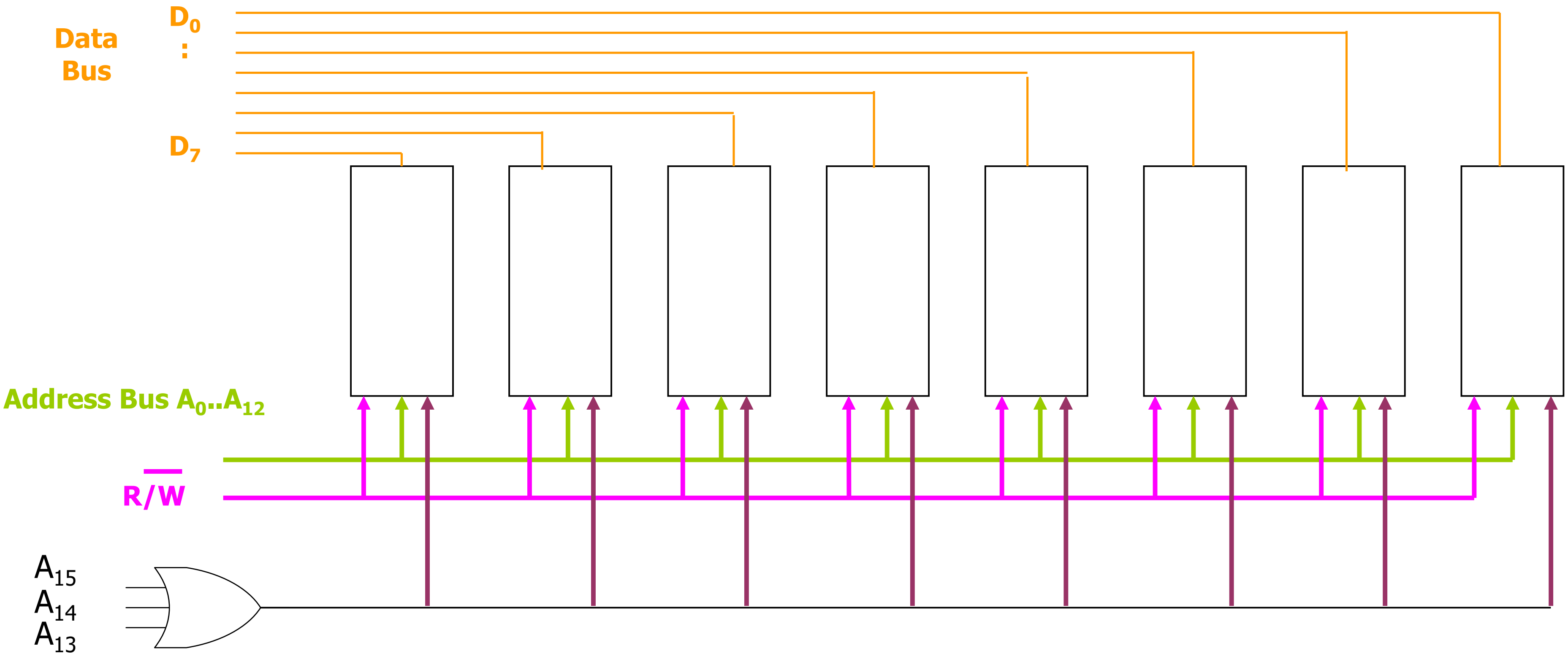




4164 DRAM: 64K x 1 Bit
(A0..A15)
8x 4164DRAM for 64K RAM

x1 bit Memory Organization

Example: Organize 8Kx1 memory chips to obtain 8Kx8 memory


Data Bus
D_0
:
D_7
Address Bus $A_0..A_{12}$
$R/\overline{W}$
A_{15}
A_{14}
A_{13}