8085 Memory Interfacing

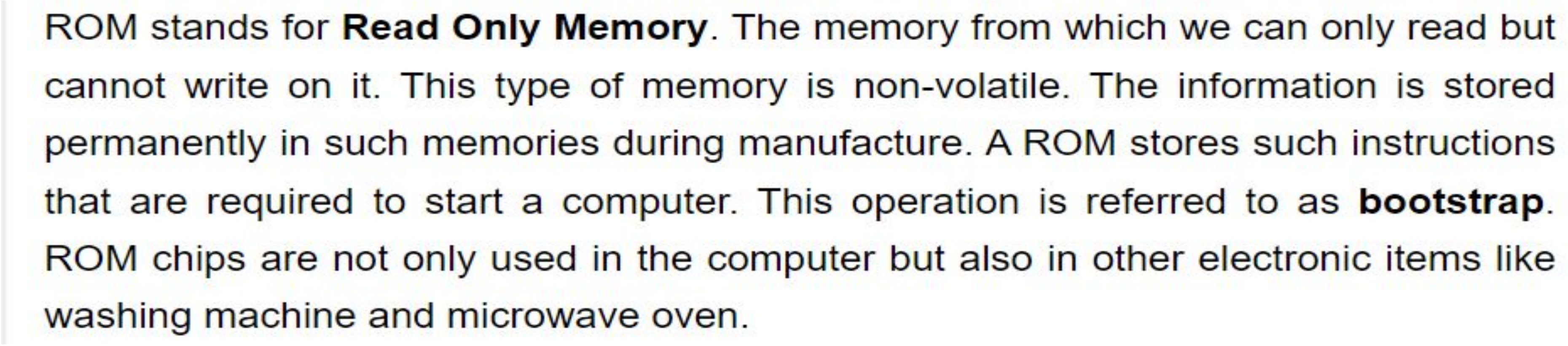
Prepared by Dr. Hema N

Types of Memory

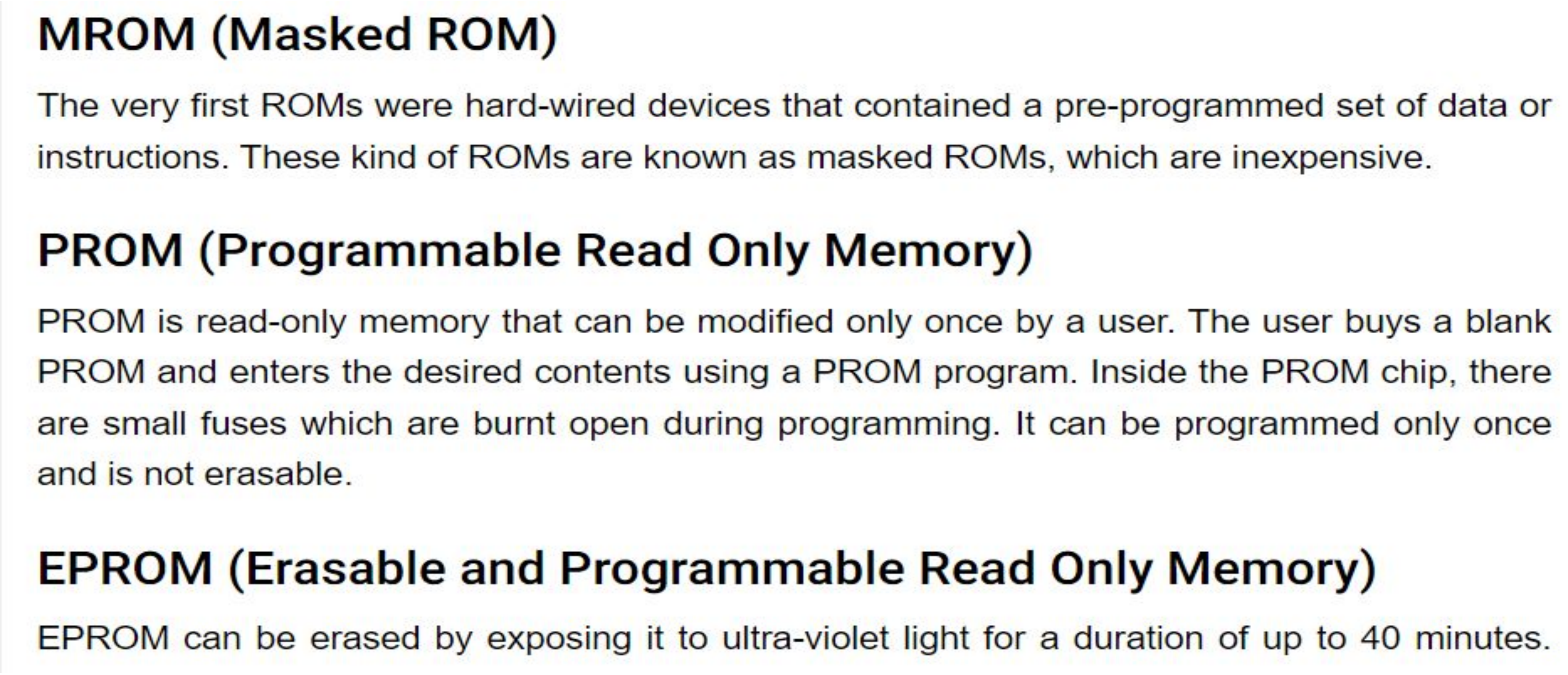
- ROM
  - MROM
  - PROM
  - EPROM
  - EEPROM
- RAM
  - SRAM
  - DRAM

https://www.tutorialspoint.com/computer_fundamentals/computer_rom_.htm

Types of Memory



ROM stands for **Read Only Memory**. The memory from which we can only read but cannot write on it. This type of memory is non-volatile. The information is stored permanently in such memories during manufacture. A ROM stores such instructions that are required to start a computer. This operation is referred to as **bootstrap**. ROM chips are not only used in the computer but also in other electronic items like washing machine and microwave oven.



MROM (Masked ROM)

The very first ROMs were hard-wired devices that contained a pre-programmed set of data or instructions. These kind of ROMs are known as masked ROMs, which are inexpensive.

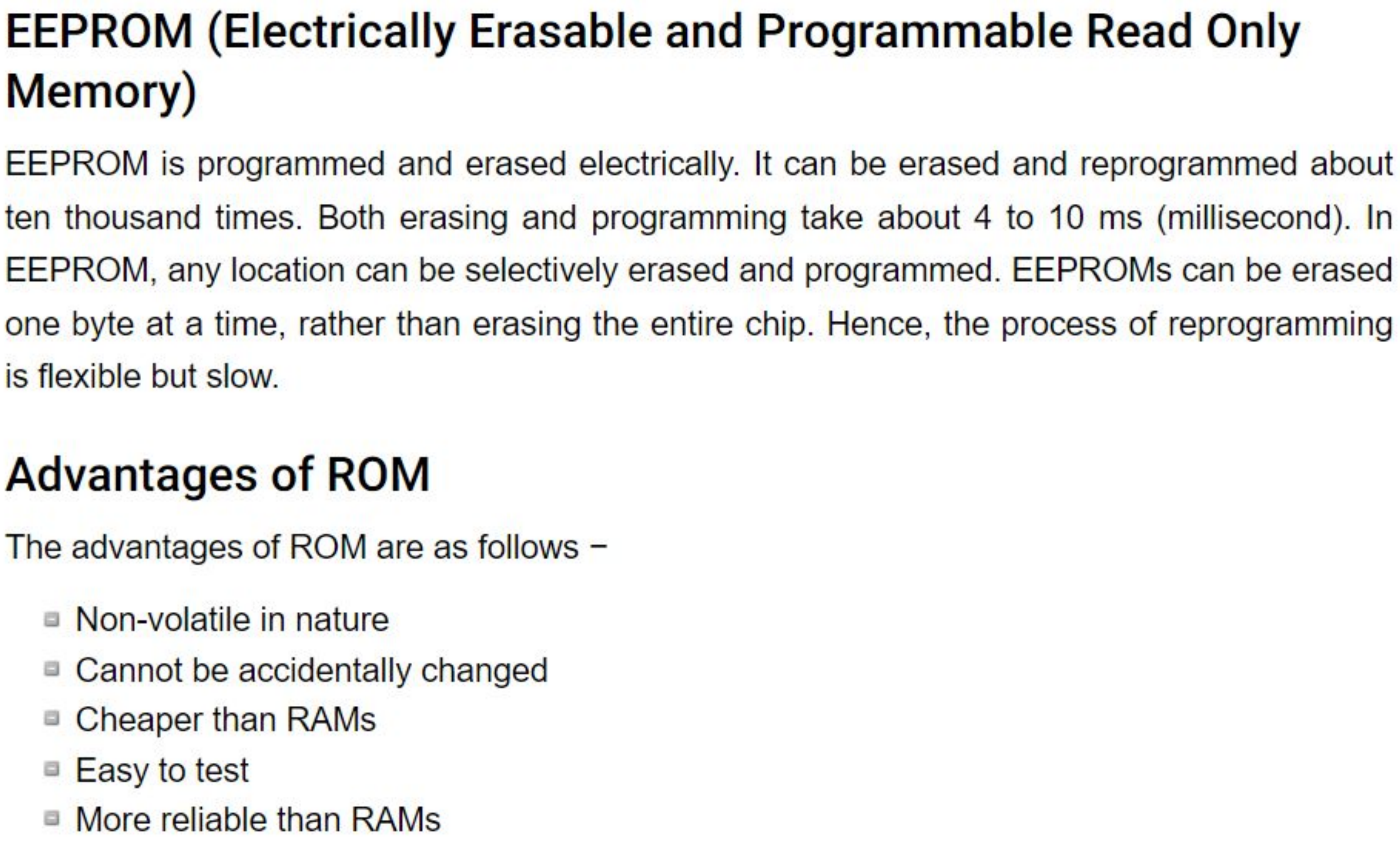
PROM (Programmable Read Only Memory)

PROM is read-only memory that can be modified only once by a user. The user buys a blank PROM and enters the desired contents using a PROM program. Inside the PROM chip, there are small fuses which are burnt open during programming. It can be programmed only once and is not erasable.

EPROM (Erasable and Programmable Read Only Memory)

EPROM can be erased by exposing it to ultra-violet light for a duration of up to 40 minutes.

Types of Memory



EEPROM (Electrically Erasable and Programmable Read Only Memory)

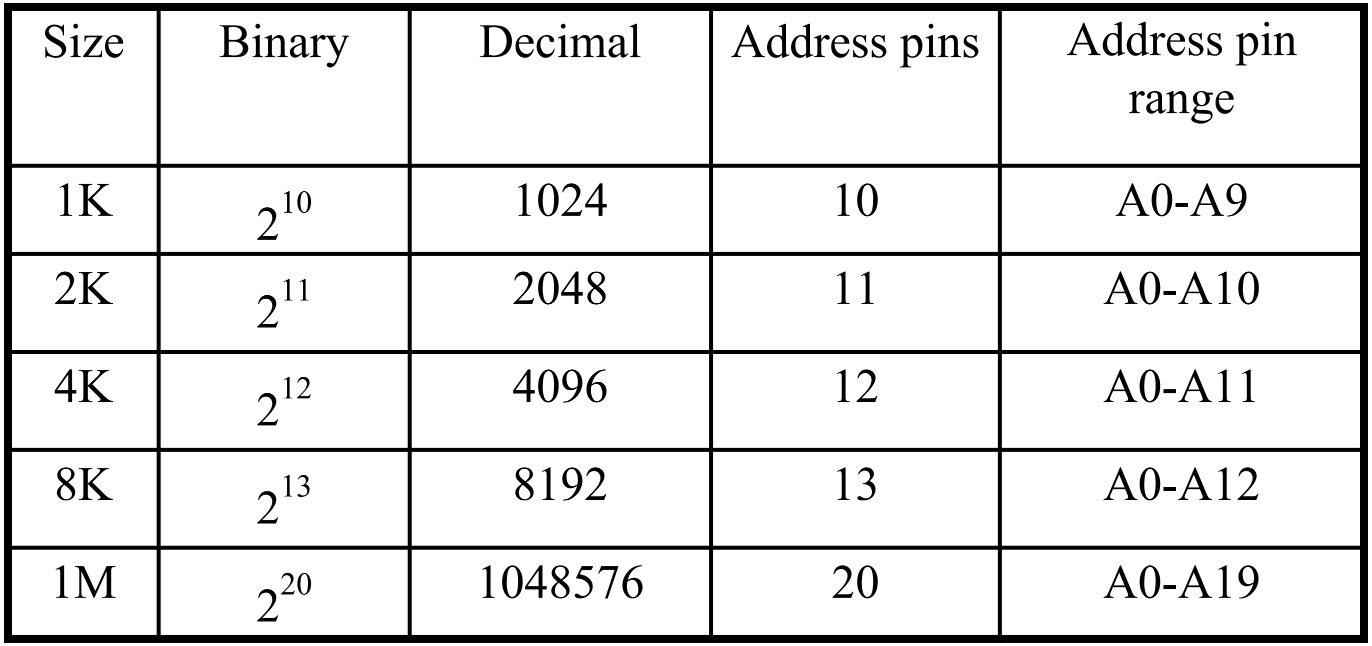
EEPROM is programmed and erased electrically. It can be erased and reprogrammed about ten thousand times. Both erasing and programming take about 4 to 10 ms (millisecond). In EEPROM, any location can be selectively erased and programmed. EEPROMs can be erased one byte at a time, rather than erasing the entire chip. Hence, the process of reprogramming is flexible but slow.

Advantages of ROM

The advantages of ROM are as follows −

- Non-volatile in nature
- Cannot be accidentally changed
- Cheaper than RAMs
- Easy to test
- More reliable than RAMs

Table showing the memory device size and corresponding address pins



Size	Binary	Decimal	Address pins	Address pin range
1K	2^{10}	1024	10	A0-A9
2K	2^{11}	2048	11	A0-A10
4K	2^{12}	4096	12	A0-A11
8K	2^{13}	8192	13	A0-A12
1M	2^{20}	1048576	20	A0-A19

Table showing the address range [starting address – ending address]

Size	Hex	Example Starting Address	End Address
1K	400H	10000H	103FFH
4K	1000H	14000H	14FFFH
64K	10000H	30000H	3FFFFH

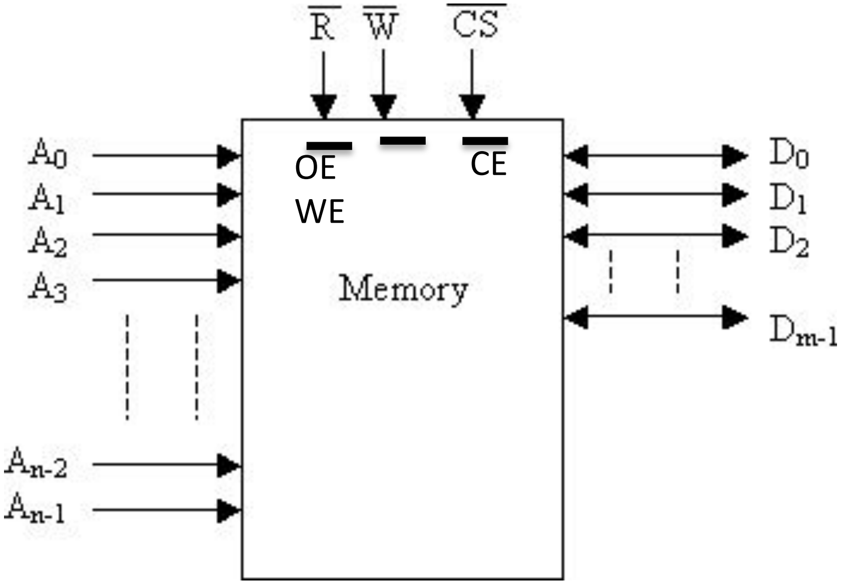
Catalog listing of memory devices

Catalog listing	Number of memory location	Bits per mem. Loc	Device name
1K x 8	1K	8	8K
16K x 1	16K	1	16K
64K x 4	64K	4	256K

Table showing the EPROM part numbers and their details

EPROM number	Details	Number of Mem. Locations	Address pins	Bits per mem. Loc	Data pins
2704	512 x 8	512	9	8	8
2708	1K x 8	1K	10	8	8
2716	2K x 8	2K	11	8	8
2732	4K x 8	4K	12	8	8
2764	8K x 8	8K	13	8	8
27256	32K x 8	32K	15	8	8
27512	64K x 8	64K	16	8	8
271024	128K x 8	128K	17	8	8

Memory Chip structure

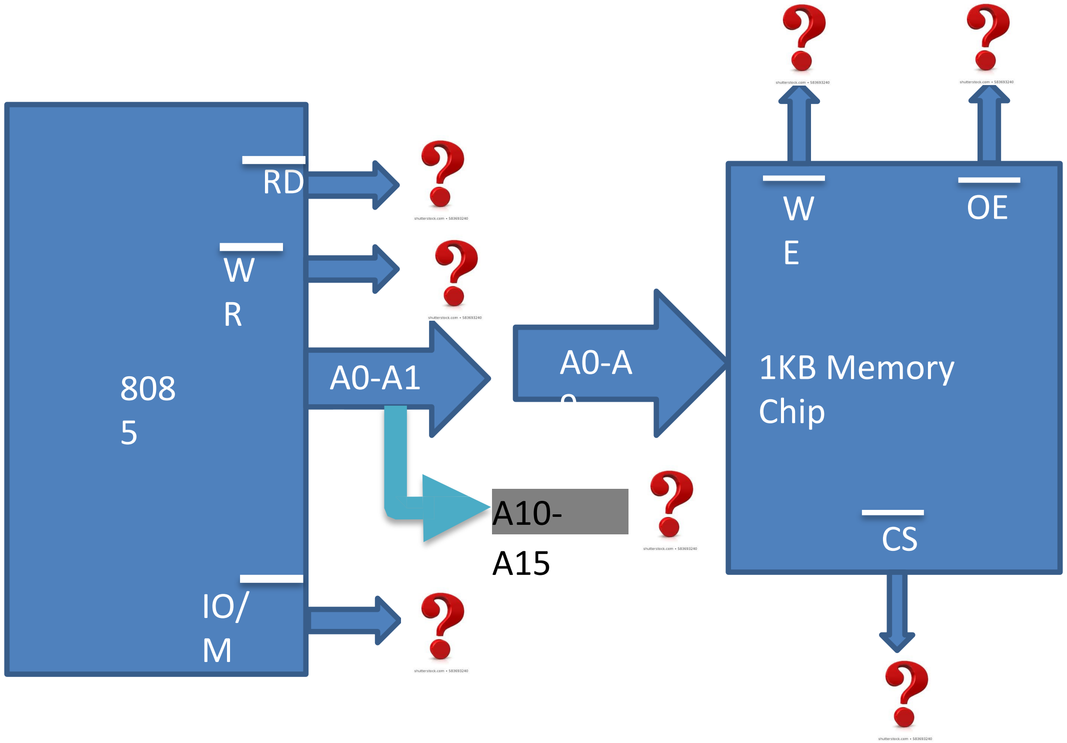


- It has n ***input*** address lines, therefore, the size of the memory is 2^n.
- It has **m** data lines, and therefore can store m bits of data for each address.
- The main lines of control for the memory are ^OE, ^WE, and ^CS.
- When ^OE is low, the processor is reading from the memory and the memory should be outputting data on its data lines.
- When ^WE is low, then the processor is writing data and the memory should be inputting data to store in the memory cells.
- If ^CS or ^CE or ^S is high, the data lines of the chip are disabled. It is if the chip is not even connected.
- When ^CS is low, the data lines are enabled for input or output based on the state of ^OE and ^WE.

http://faculty.etsu.edu/tarnoff/hmwk2150/chipslct/chipslct.html

Decoder Logic in memory Interface

- 1KB memory chip interface with 8085
- For the interfacing, theremaining address pins must be decoded using thetechnique called "**Address Decoding**".



Techniques for decoding

i)Logic Gate Decoder (NOT gate decoder)

ii)Logic Gate Decoder (Simple NAND Gate decoder)

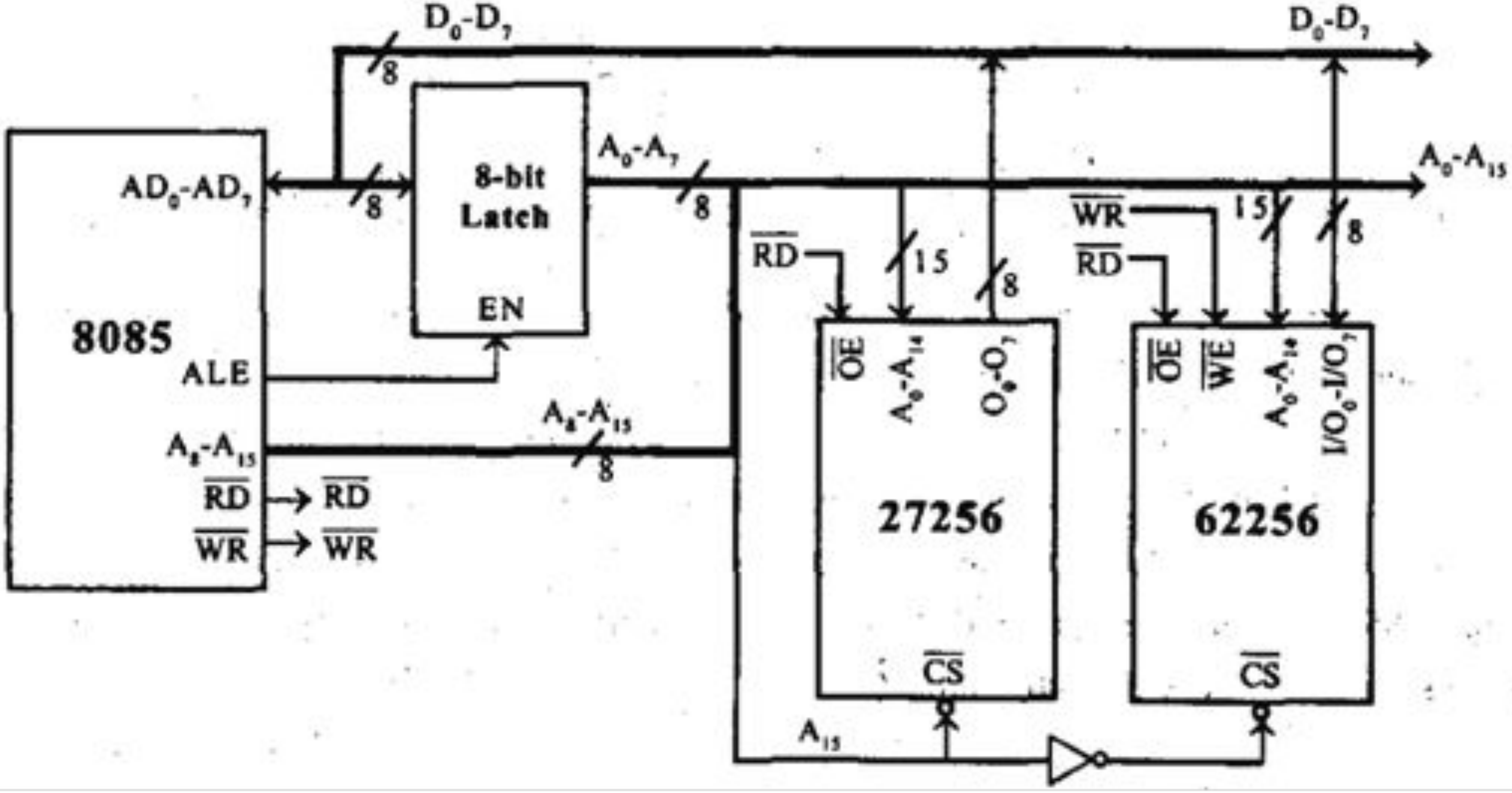
iii) Line Decoder

- The 1-to-2 line decoder
- The dual 2-to-4 line decoder
- The 3-to-8 line decoder

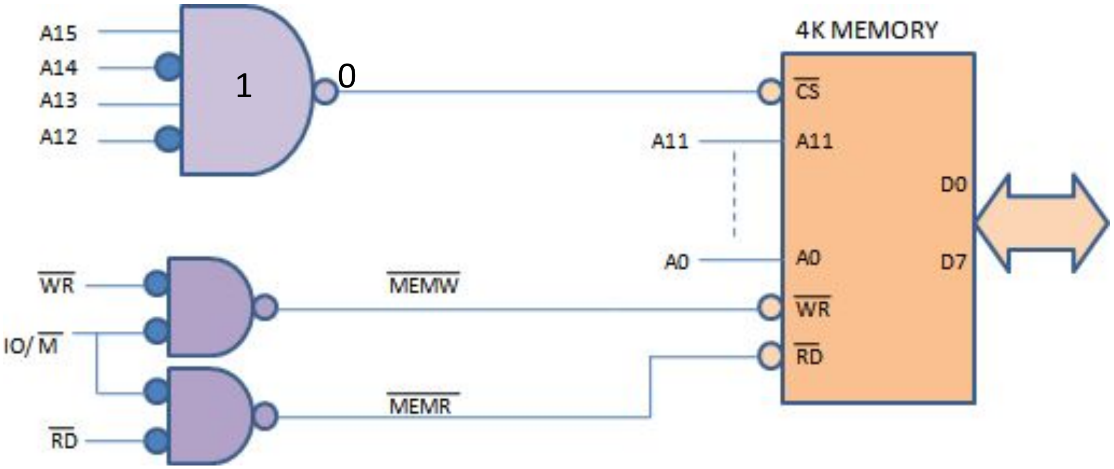
https://deeprajbhujel.blogspot.com/2015/09/how-do-you-interface-8085-microprocessor.html

NOT gate decoder

- **Consider a system in which the available 64kb memory space is equally divided between EPROM and RAM. Interface the EPROM and RAM with 8085 processor.**
  - **Implement 32kb memory capacity of EPROM using single IC 27256.**
  - **32kb RAM capacity is implemented using single IC 62256.**
  - **The 32kb memory requires 15 address lines and so the address lines A0 - A14 of the processor are connected to 15 address pins of both EPROM and RAM.**
  - **The unused address line A15 is used as to chip select. If A15 is 1, it select RAM and If A15 is 0, it select EPROM.**
- **Inverter is used for selecting the memory.**
- **The memory used is both RAM and EPROM, so the low RD and WR pins of processor are connected to low WE and OE pins of memory respectively.**
- **The address range of EPROM will be 0000H to 7FFFH and that of RAM will be 7FFFH to FFFFH.**

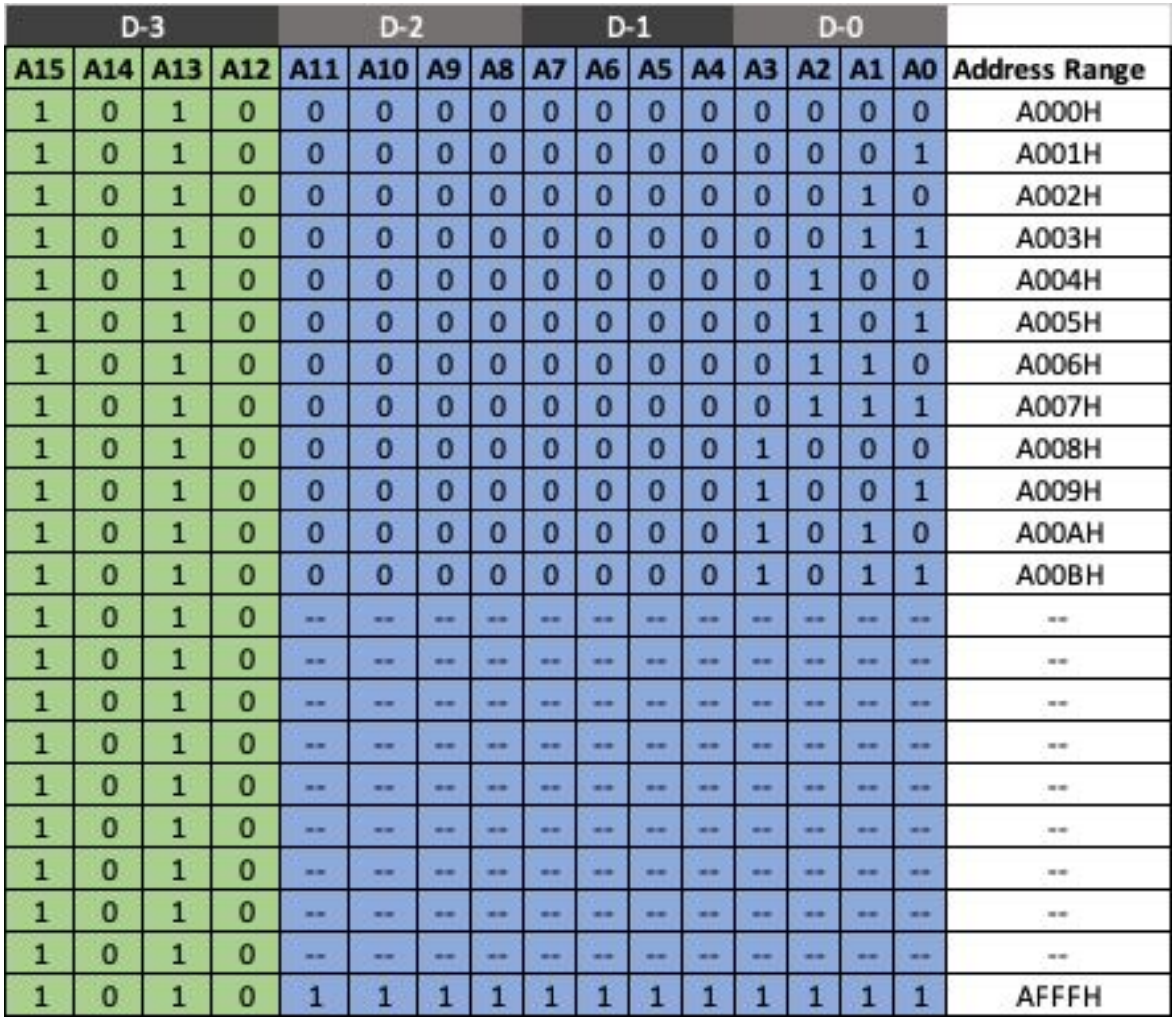

D0-D7
8
D0-D7
8085
AD0-AD7
8
8-bit Latch
EN
A0-A7
8
A0-A15
ALE
A8-A15
A8-A15
8
RD → RD
WR → WR
RD
15
8
OE
A0-A14
O0-O7
27256
CS
WR
RD
15
8
OE
WE
A0-A14
I/O0-I/O7
62256
CS
A15

NAND gate Decoder logic Interfacing



- 4KB memory Interface using NAND gate
- Address Range as follows:

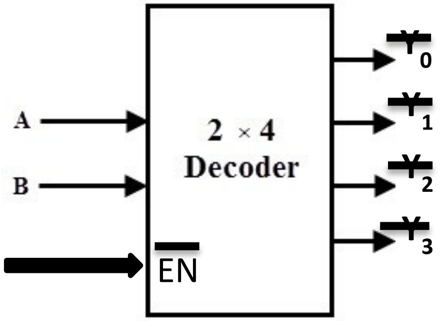
https://deeprajbhujel.blogspot.com/2015/09/how-do-you- interface-8085-microprocessor.html



D-3				D-2				D-1				D-0				
A15	**A14**	**A13**	**A12**	**A11**	**A10**	**A9**	**A8**	**A7**	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**	**Address Range**
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	A000H
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	A001H
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	A002H
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	A003H
1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	A004H
1	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	A005H
1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	A006H
1	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	A007H
1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	A008H
1	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	A009H
1	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	A00AH
1	0	1	0	0	0	0	0	0	0	0	0	1	0	1	1	A00BH
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
1	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	AFFFH

2 to 4 decoder(74LS139) logic

Memory Interface



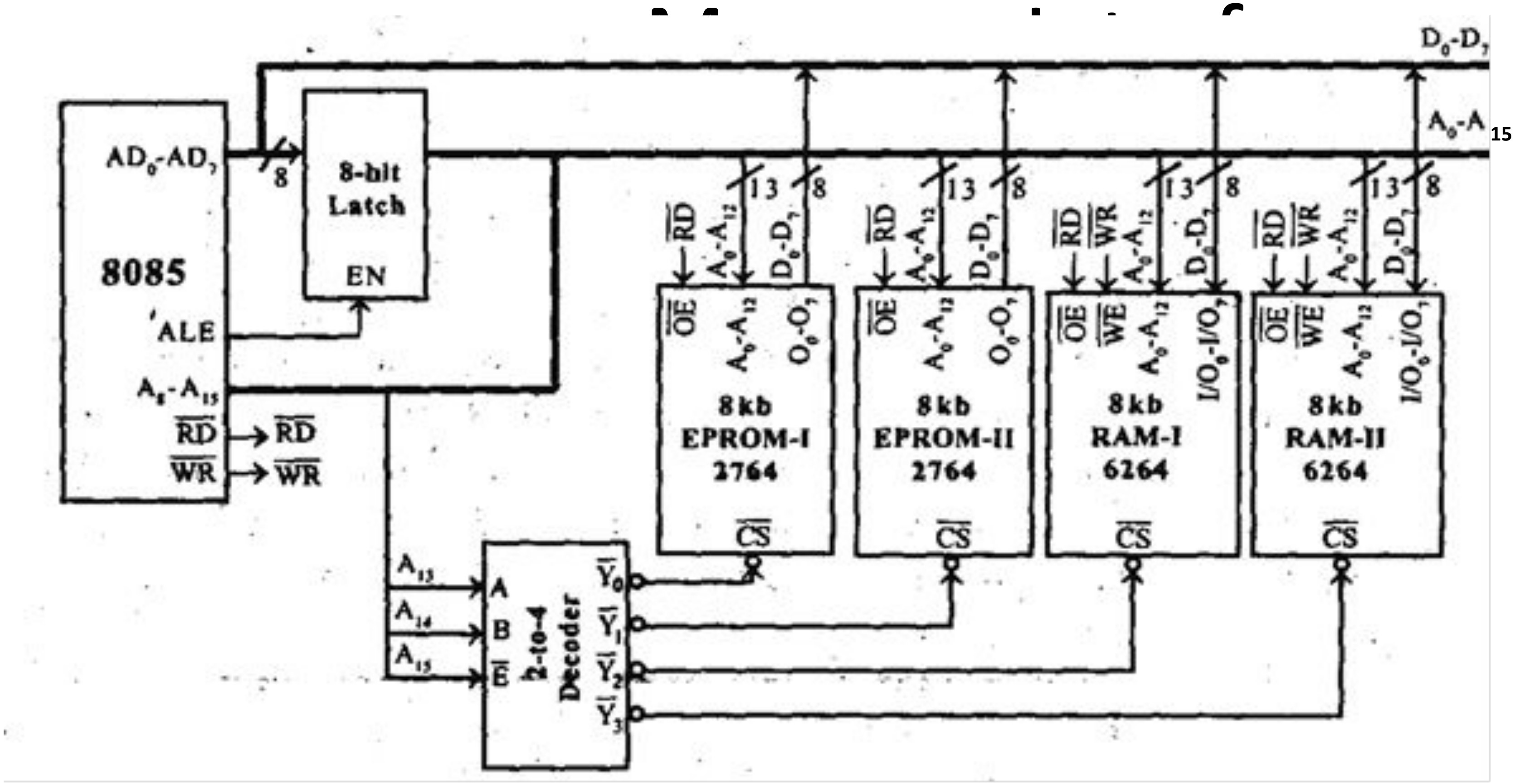
Inputs			Output			
$\overline{E}$	B	A			$\overline{Y_2}$	$\overline{Y_3}$
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	X	X	1	1	1	1

- In 2x4 decoder, For a given input, the outputs $\overline{Y_0}$ through $\overline{Y_3}$ are active high if enable input EN is active high ($\overline{EN}$ = 1).
- When ($\overline{EN}$=0) and both inputs A and B are low (or A= B= 0), the output $\overline{Y_0}$ will be active or LOW and all other outputs will be high.

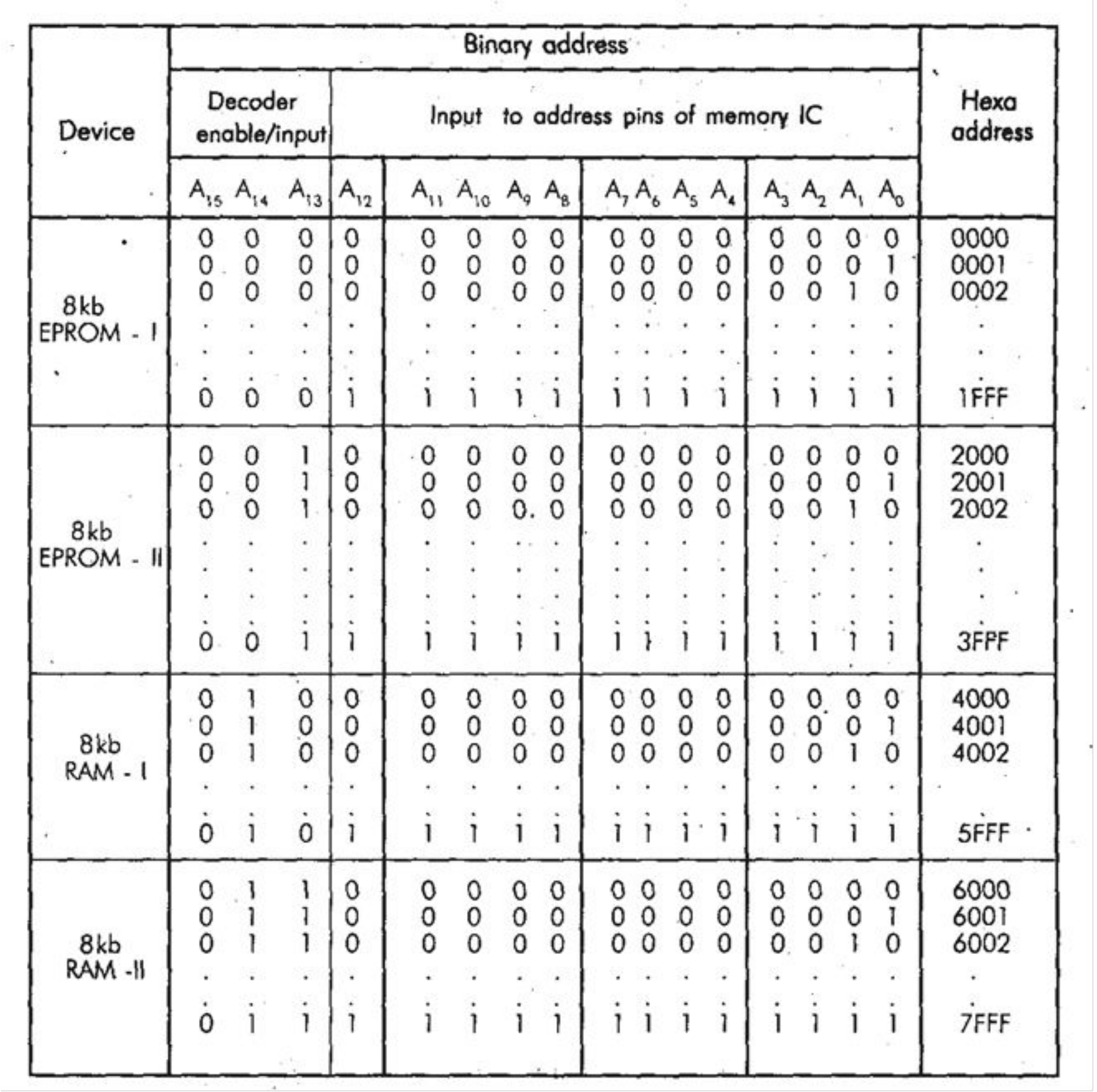
2 to 4 decoder logic Memory Interface

- **Consider a system in which 32KB memory space is implemented using four numbers of 8KB memory. Interface the EPROM and RAM with 8085 processor.**
  - **The total memory capacity is 32KB. So, let two number of 8KB memory be EPROM and the remaining two numbers be RAM.**
  - **Each 8KB memory requires 13 address lines and so the address lines A0- A12 of the processor are connected to 13 address pins of all the memory.**
  - **The address lines and A13 - A14 can be decoded using a 2-to-4 decoder to generate four chip select signals.**
  - **These four chip select signals can be used to select one of the four memory chips at any one time.**
  - **The address line A15 is used as enable for decoder.**
  - **The simplified schematic memory organization is shown.**

2 to 4 decoder logic



2 to 4 decoder logic Memory



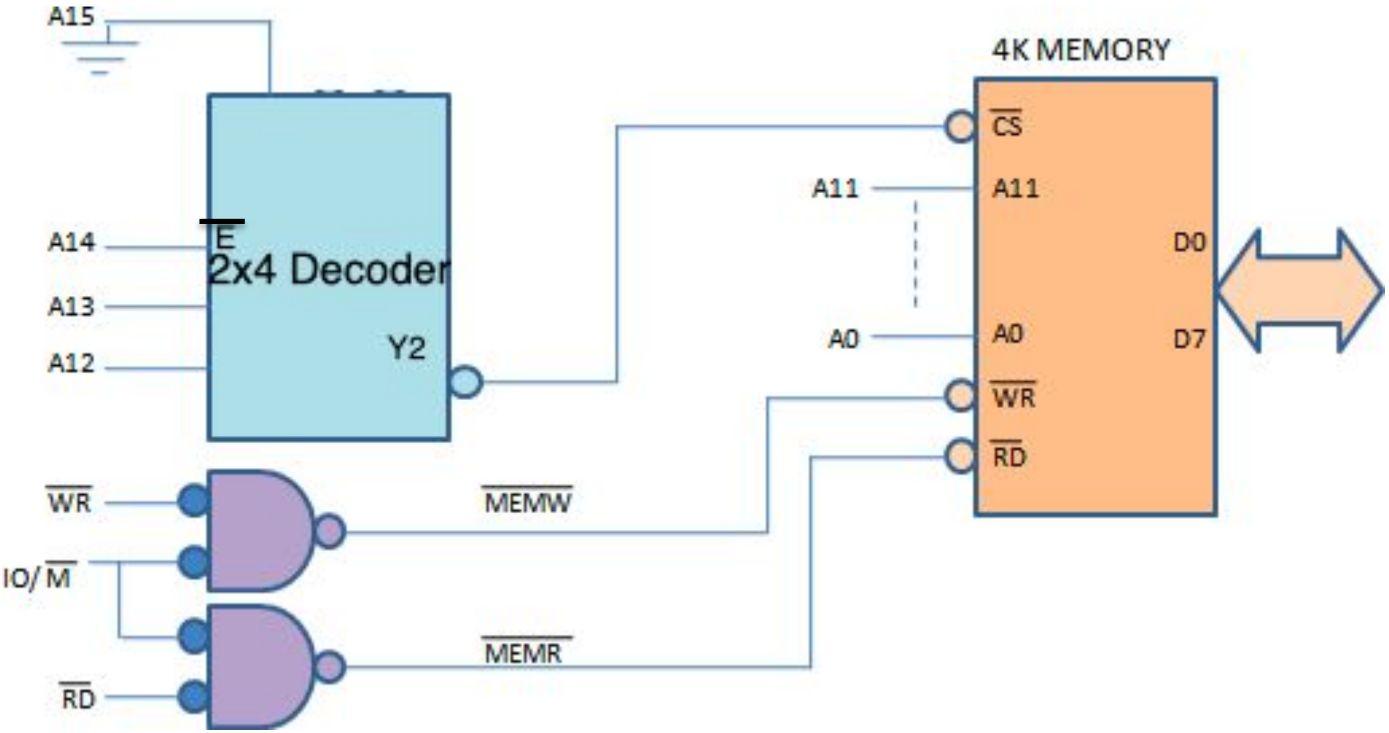
Device	Binary address																Hexa address
	Decoder enable/input			Input to address pins of memory IC													
	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
8kb EPROM - I	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0002
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1FFF
8kb EPROM - II	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2000
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	2001
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	2002
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFF
8kb RAM - I	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4000
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4001
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	4002
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	
	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	5FFF
8kb RAM -II	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	6000
	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	6001
	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	6002
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.
	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	7FFF

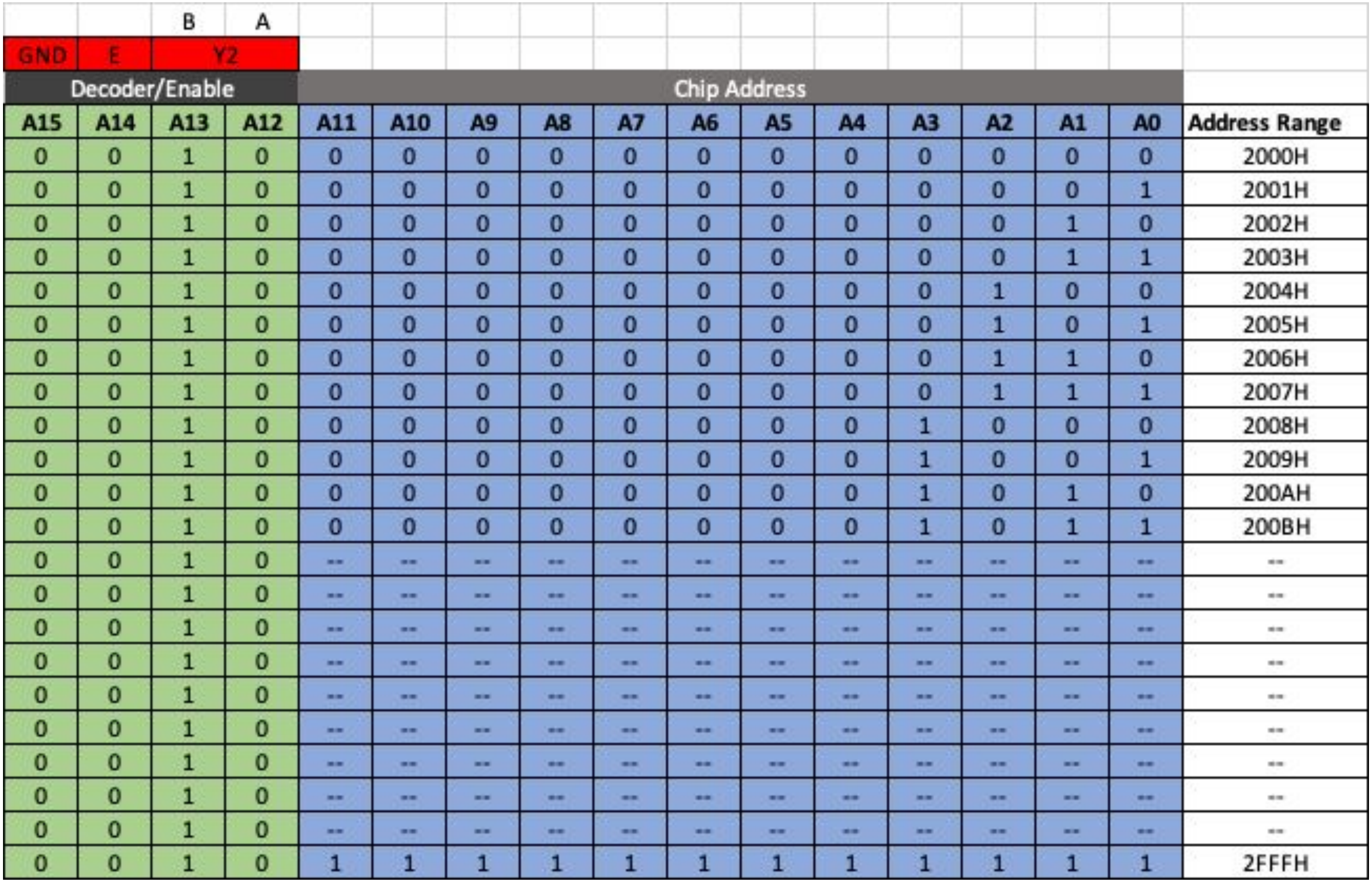
The address allotted to each memory IC is shown in following table.

http://www.8085projects.info/Examples-of-Memory-Interfacing-Contd-Page3.html

2 to 4 decoder logic

Memory Interface

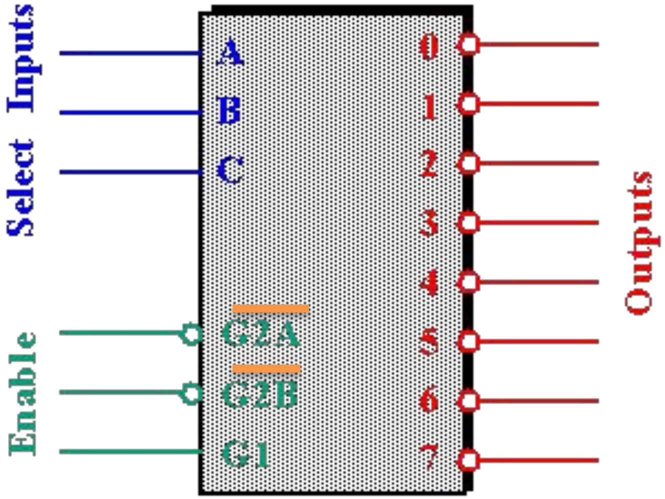


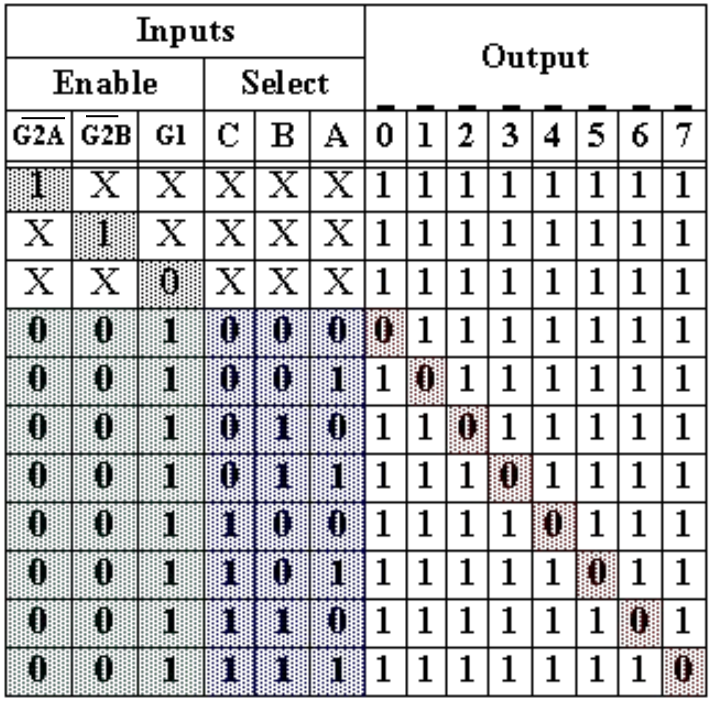


		B	A													
GND	E	Y2														
Decoder/Enable				Chip Address												
A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Address Range
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2000H
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	2001H
0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	2002H
0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	2003H
0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	2004H
0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	1	2005H
0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	0	2006H
0	0	1	0	0	0	0	0	0	0	0	0	0	1	1	1	2007H
0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	2008H
0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	2009H
0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	0	200AH
0	0	1	0	0	0	0	0	0	0	0	0	1	0	1	1	200BH
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	--	--	--	--	--	--	--	--	--	--	--	--	--
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	2FFFH

3 to 8 decoder logic interface

- *The 3-to-8 Line Decoder (74LS138)*



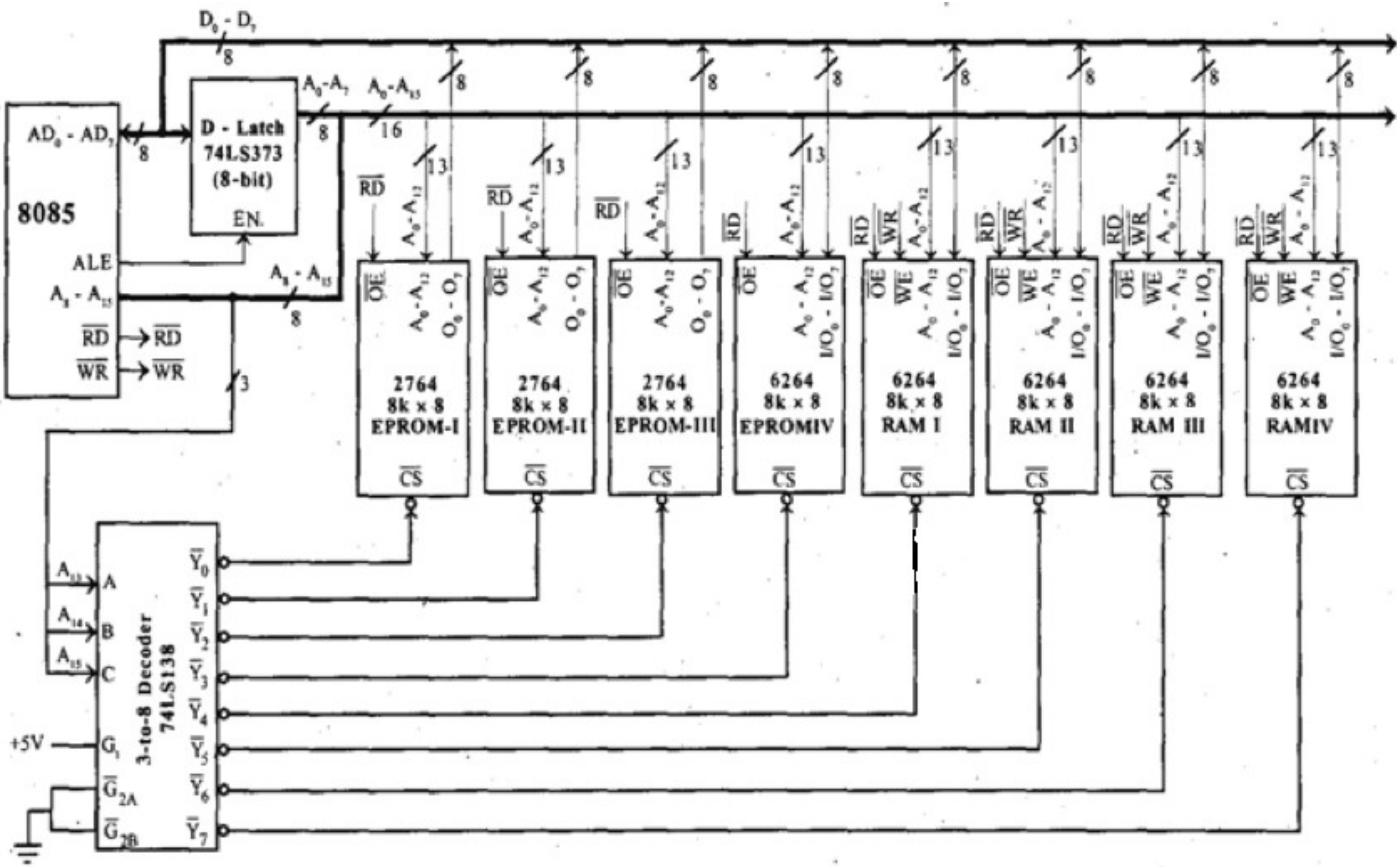


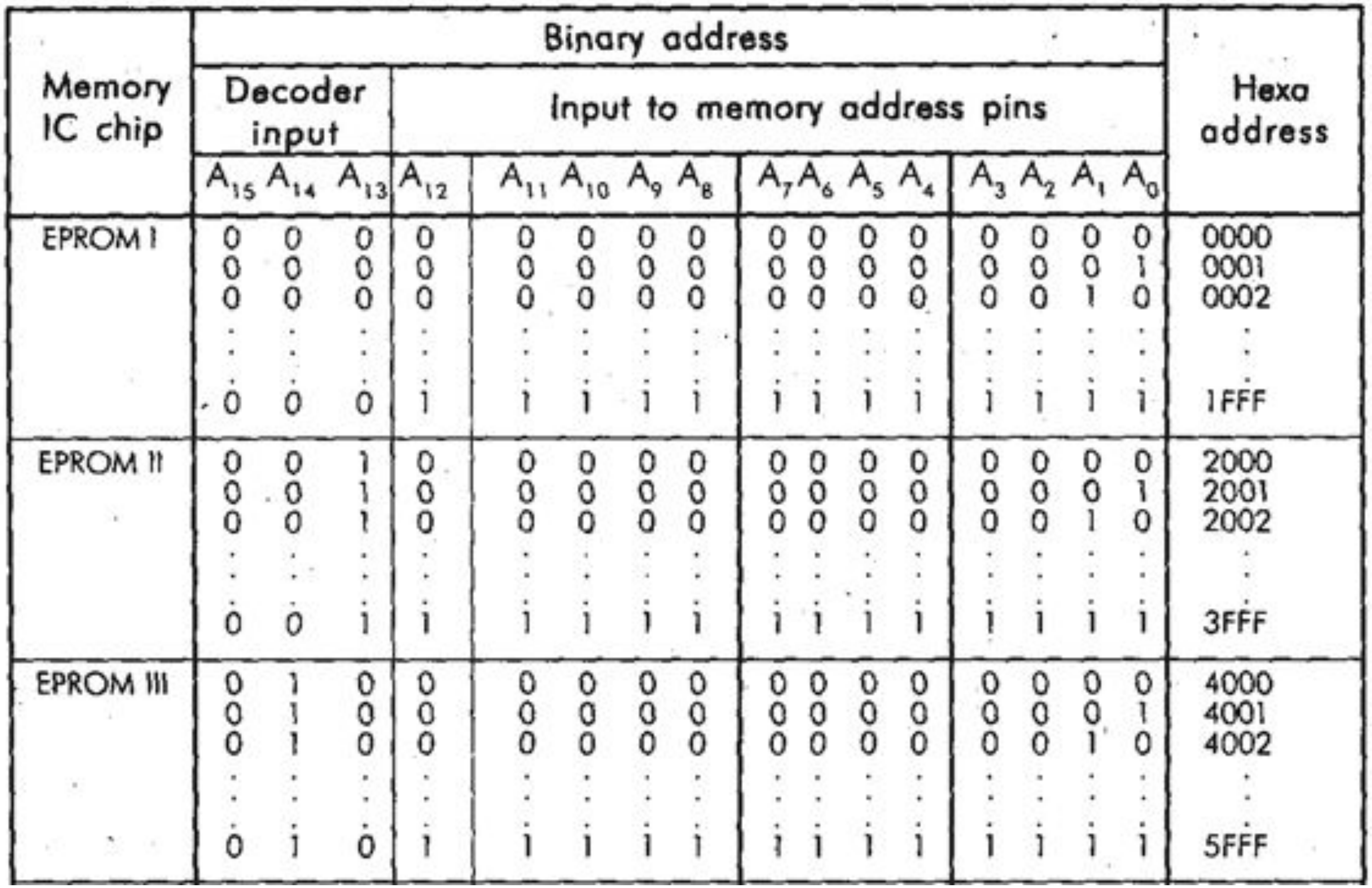
Inputs						Output							
Enable			Select										
$\overline{G2A}$	$\overline{G2B}$	G1	C	B	A	$\overline{0}$	$\overline{1}$	$\overline{2}$	$\overline{3}$	$\overline{4}$	$\overline{5}$	$\overline{6}$	$\overline{7}$
1	X	X	X	X	X	1	1	1	1	1	1	1	1
X	1	X	X	X	X	1	1	1	1	1	1	1	1
X	X	0	X	X	X	1	1	1	1	1	1	1	1
0	0	1	0	0	0	0	1	1	1	1	1	1	1
0	0	1	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	0	1	1	0	1	1	1	1	1
0	0	1	0	1	1	1	1	1	0	1	1	1	1
0	0	1	1	0	0	1	1	1	1	0	1	1	1
0	0	1	1	0	1	1	1	1	1	1	0	1	1
0	0	1	1	1	0	1	1	1	1	1	1	0	1
0	0	1	1	1	1	1	1	1	1	1	1	1	0

3 to 8 decoder logic interface

- **Consider a system in which the 64KB memory space is implemented using eight numbers of 8KB memory. Interface the EPROM and RAM with 8085 processor.**
  - **The total memory capacity is 64KB. So, let 4 numbers of 8Kb EPROM and 4 numbers of 8KB RAM.**
  - **Each 8kb memory requires 13 address lines. So the address line A0 - A12 of the processor are connected to 13address pins of all the memory chip.**
  - **The address lines A13, A14 and A15 are decoded using a 3-to-8 coder to generate eight chip select signals. These eight chip select signals can be used to select one of the eight memories at any one time.**
- **The memory interfacing is shown in following figure.**

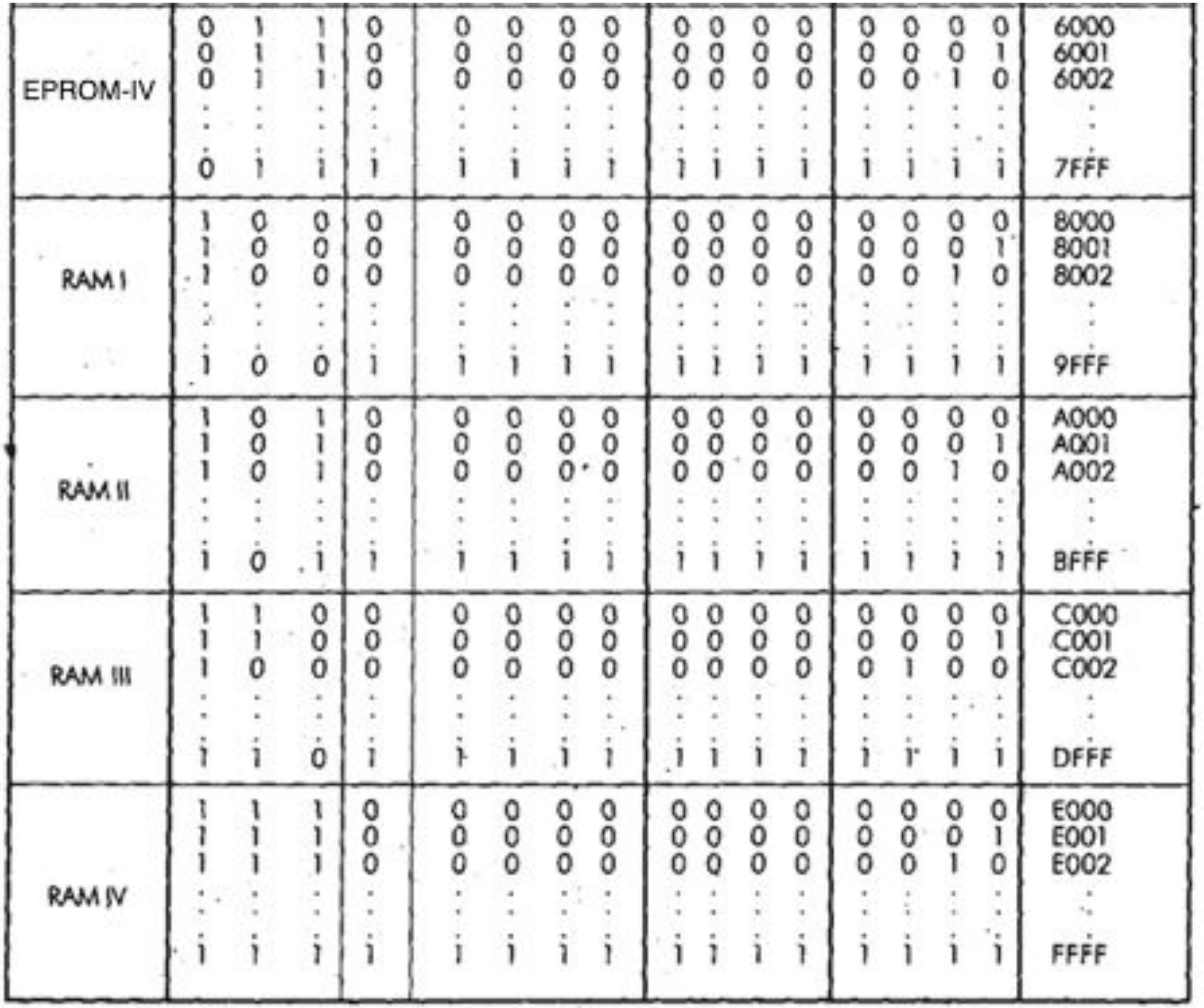
http://www.8085projects.info/Examples-of-Memory-Interfacing-Contd-Page4_.html


8085
AD0 - AD7
ALE
A8 - A15
RD
WR
D0 - D7
8
D - Latch
74LS373
(8-bit)
EN.
A0-A7
A0-A15
16
A8 - A15
13
3
OE
A0-A12
O0 - O7
2764
8k × 8
EPROM-I
EPROM-II
EPROM-III
CS
6264
EPROMIV
WE
I/O0 - I/O7
RAM I
RAM II
RAM III
RAMIV
A13
A14
A15
A
B
C
+5V
G1
G2A
G2B
3-to-8 Decoder
74LS138
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7



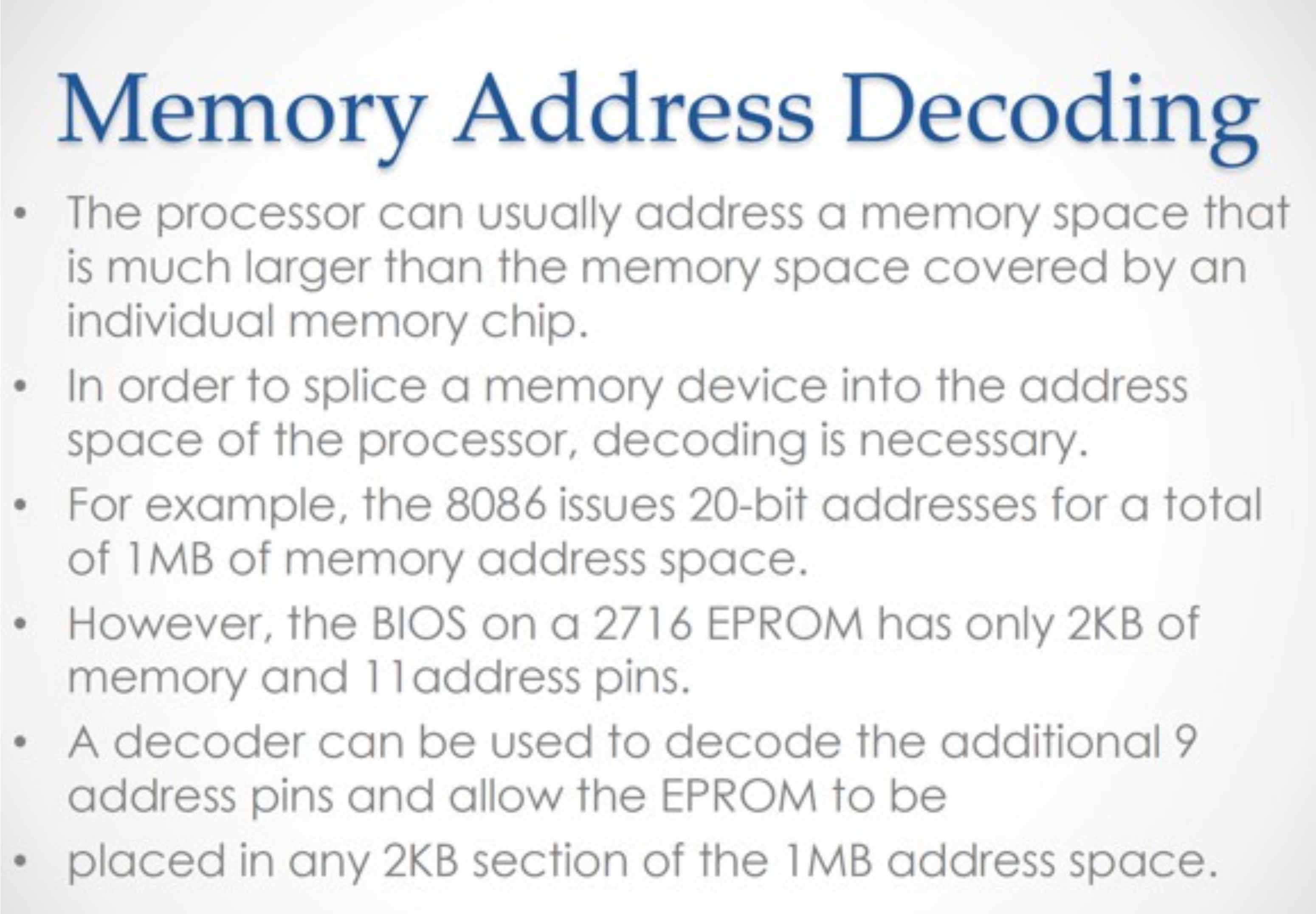
Memory IC chip	Binary address																Hexa address
	Decoder input			Input to memory address pins													
	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	
EPROM I	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0001
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1FFF
EPROM II	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2000
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	2001
	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	2002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFF
EPROM III	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4000
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	4001
	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	4002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	5FFF

https://www.slis
hare.net/Srikrisl
a Thota/8085-
interfacing-with
memory-chips-
58311824



EPROM-IV	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	6000
	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	6001
	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	6002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	7FFF
RAM I	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	8000
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	8001
	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	8002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	9FFF
RAM II	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	A000
	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	A001
	1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	A002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	BFFF
RAM III	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	C000
	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	C001
	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	C002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	DFFF
RAM IV	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	E000
	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	E001
	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	E002
	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	FFFF

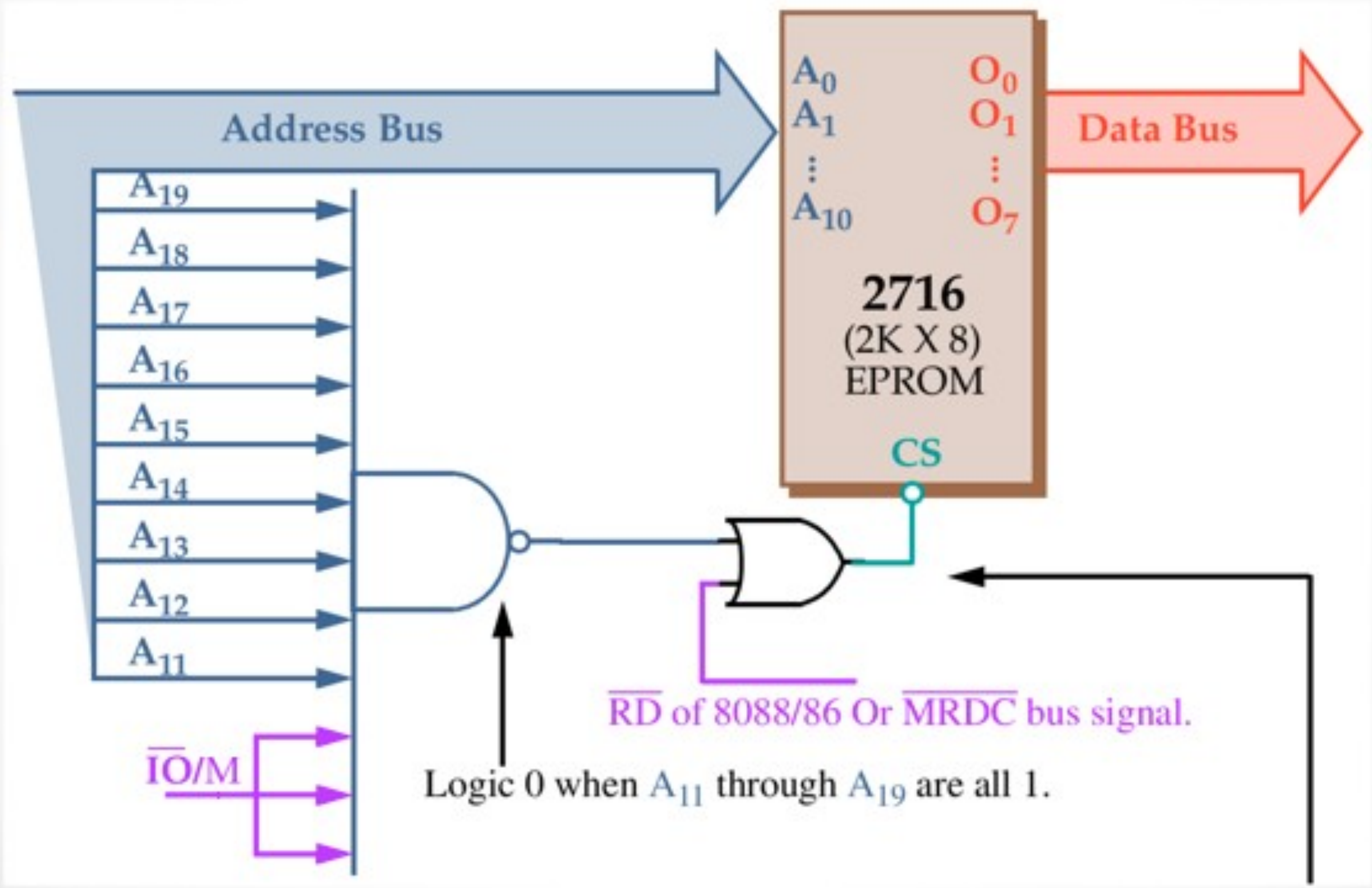
8086 Memory



Memory Address Decoding

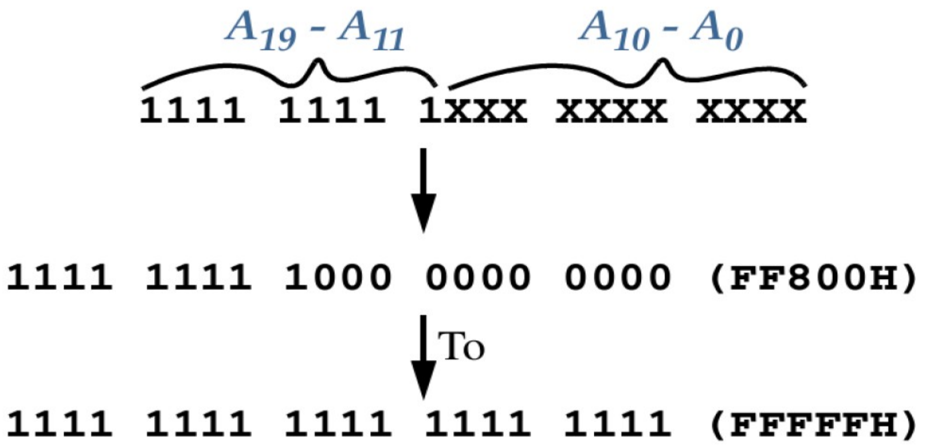
- The processor can usually address a memory space that is much larger than the memory space covered by an individual memory chip.
- In order to splice a memory device into the address space of the processor, decoding is necessary.
- For example, the 8086 issues 20-bit addresses for a total of 1MB of memory address space.
- However, the BIOS on a 2716 EPROM has only 2KB of memory and 11address pins.
- A decoder can be used to decode the additional 9 address pins and allow the EPROM to be
- placed in any 2KB section of the 1MB address space.

NAND Decoder Example



NAND Decoder Example

- To determine the address range that a device is mapped into



A_{19} - A_{11} A_{10} - A_0

1111 1111 1XXX XXXX XXXX

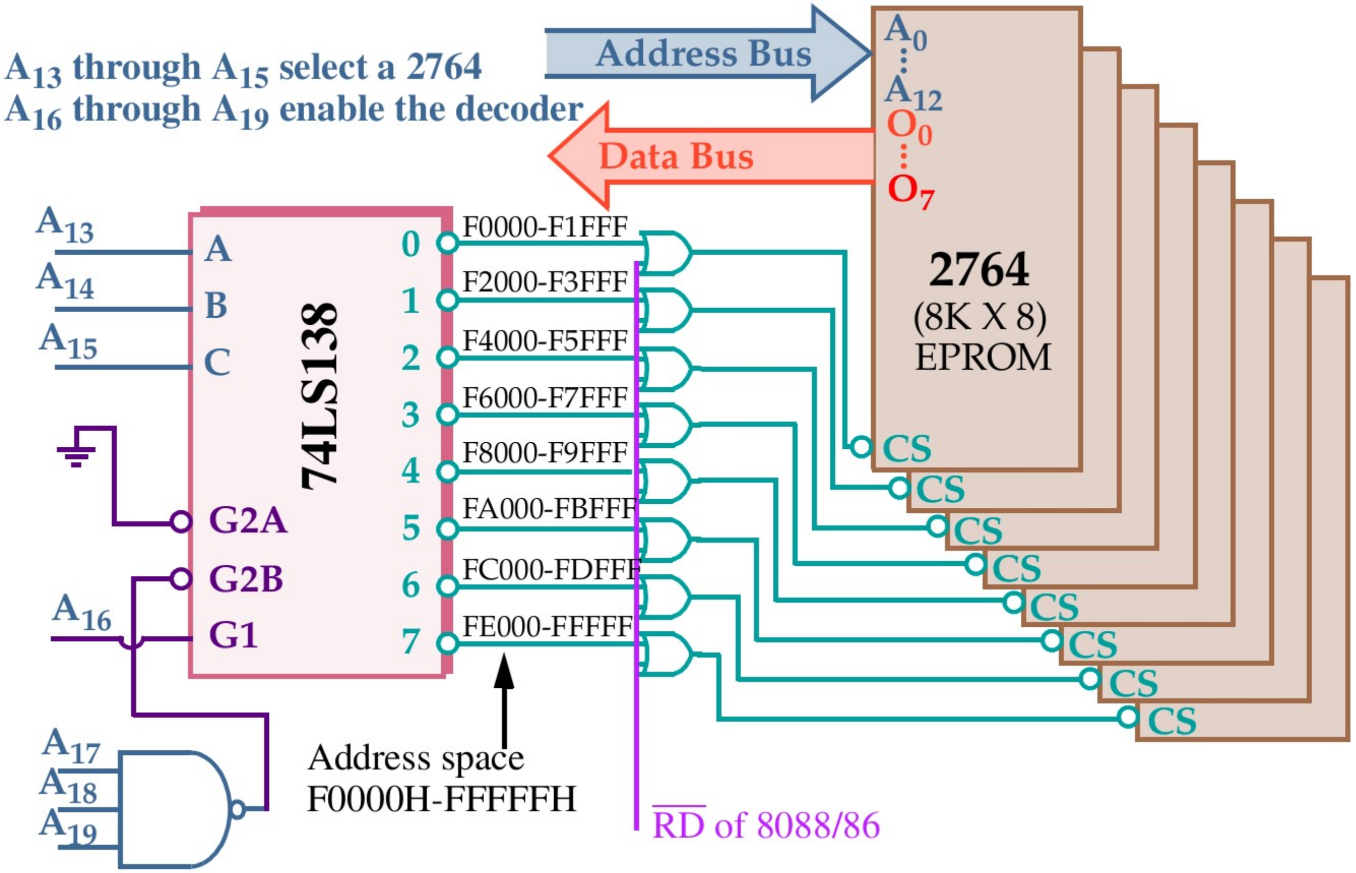
1111 1111 1000 0000 0000 (FF800H)

To

1111 1111 1111 1111 1111 (FFFFFH)

- NAND gate decoders are not often used
  - Large fan-in NAND gates are not efficient
  - Multiple NAND gate IC's might be required to perform such decoding
  - Rather the 3-to-8 Line Decoder (74LS138) is more common.

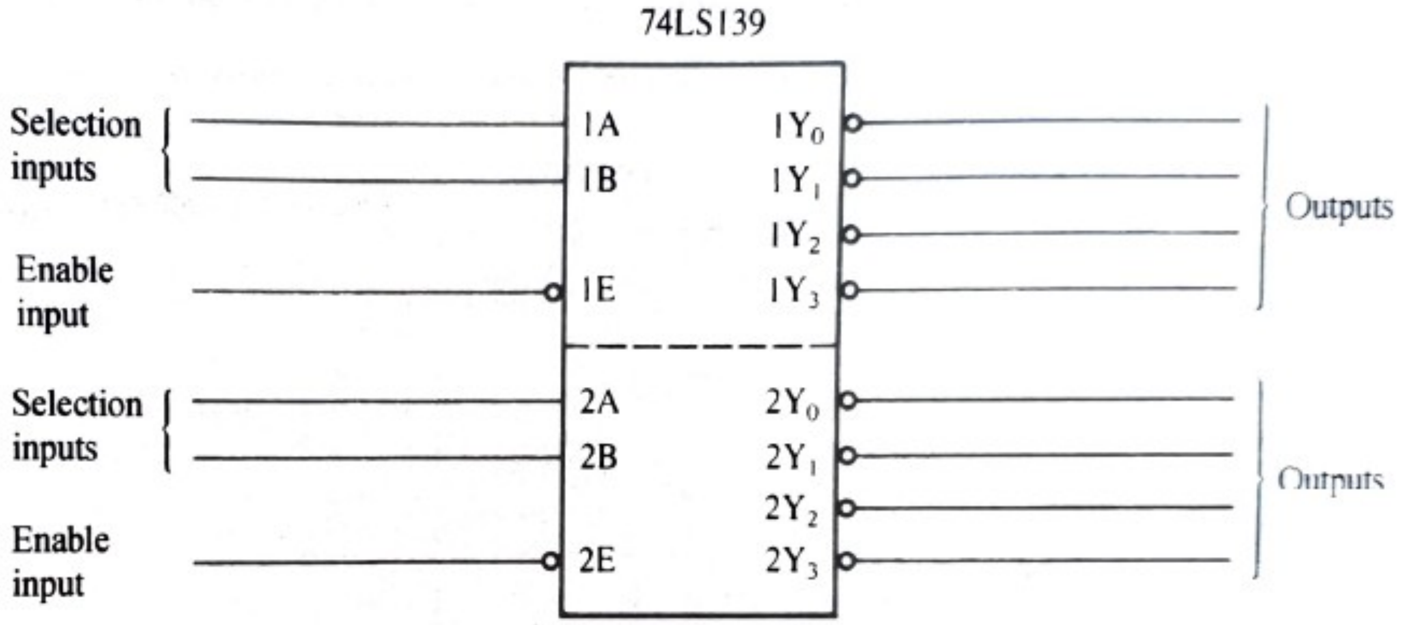
3 to 8 Line decoder with 8-bit Memory Interface



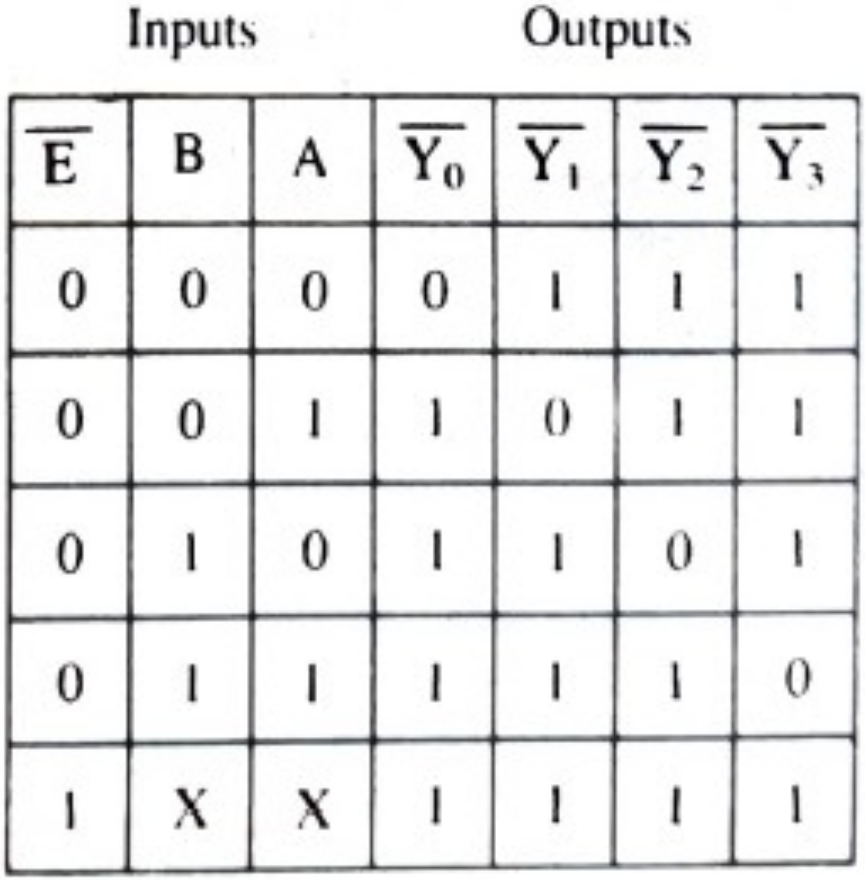
The EPROMs cover a 64KB section of memory.

Dual 2-to-4 Line Decoder

74LS139 is a dual 2-to-4 line decoder

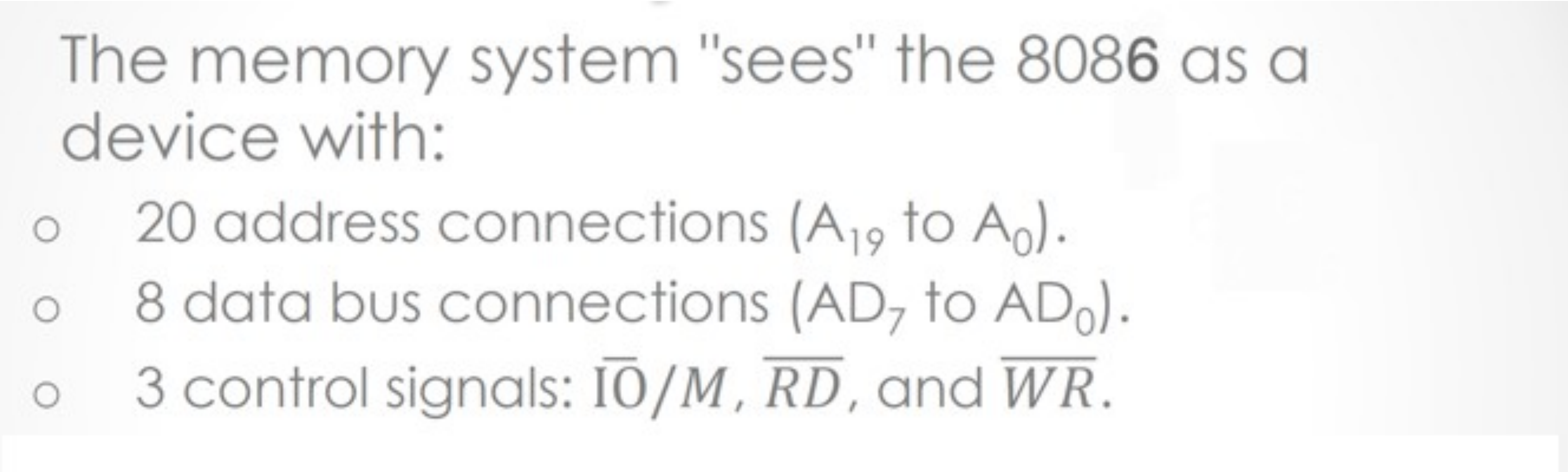


Dual 2-to-4 Line Decoder



Inputs			Outputs			
$\overline{E}$	B	A	$\overline{Y_0}$	$\overline{Y_1}$	$\overline{Y_2}$	$\overline{Y_3}$
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0
1	X	X	1	1	1	1

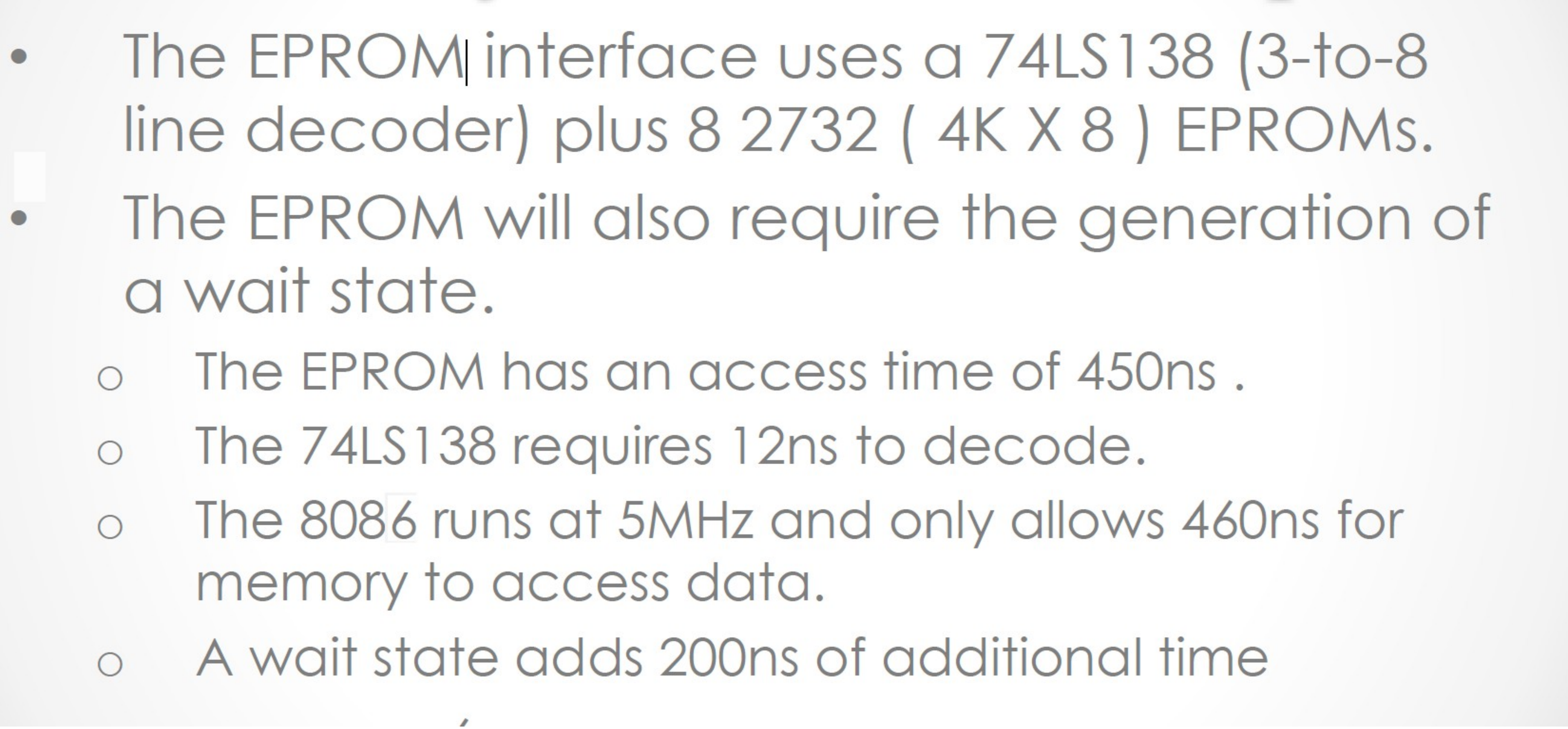
8086 8-Bit Memory Interface



The memory system "sees" the 808**6** as a device with:

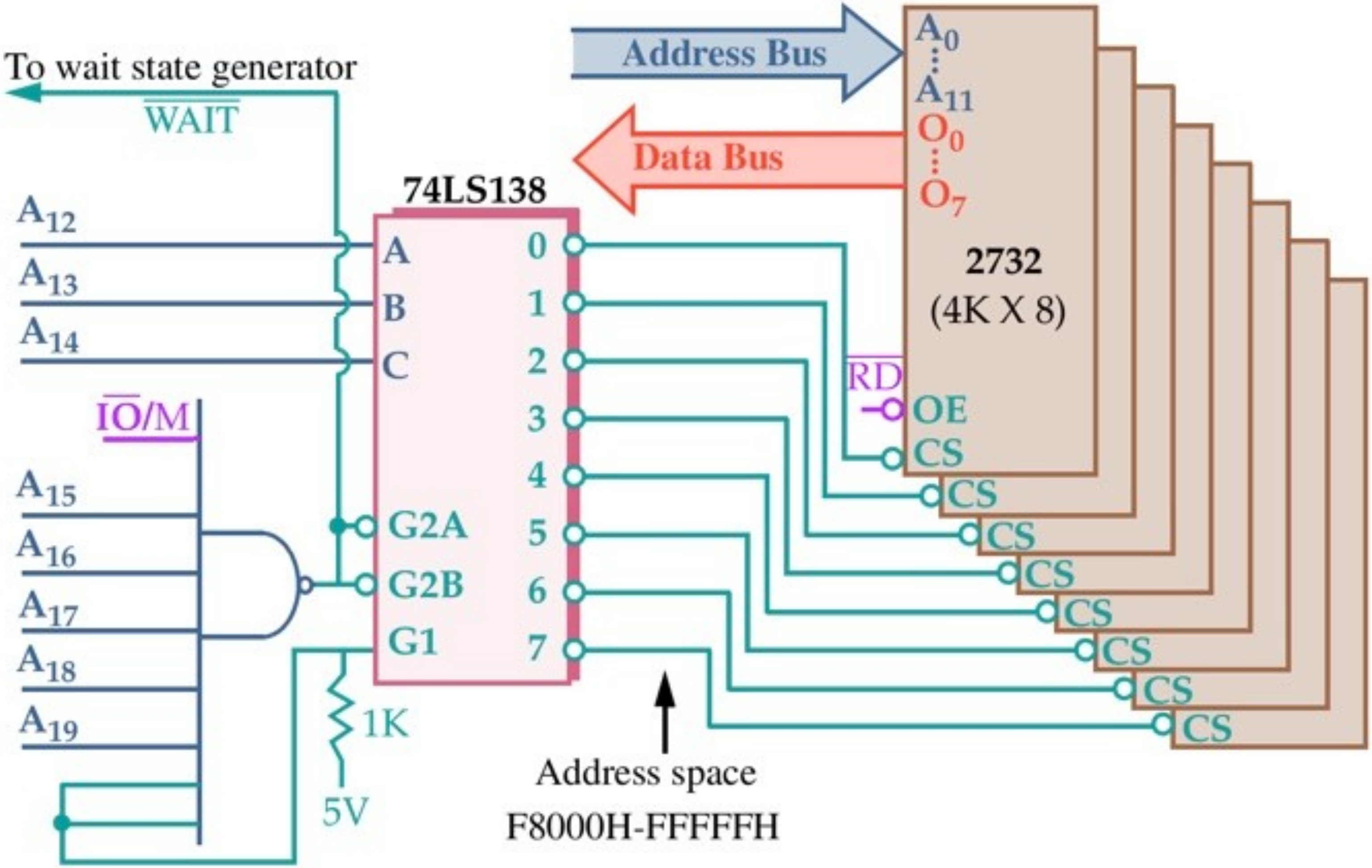
- 20 address connections (A_{19} to A_0).
- 8 data bus connections (AD_7 to AD_0).
- 3 control signals: $\overline{\mathrm{IO}}/M$, $\overline{RD}$, and $\overline{WR}$.

8086 8-Bit Memory Interface

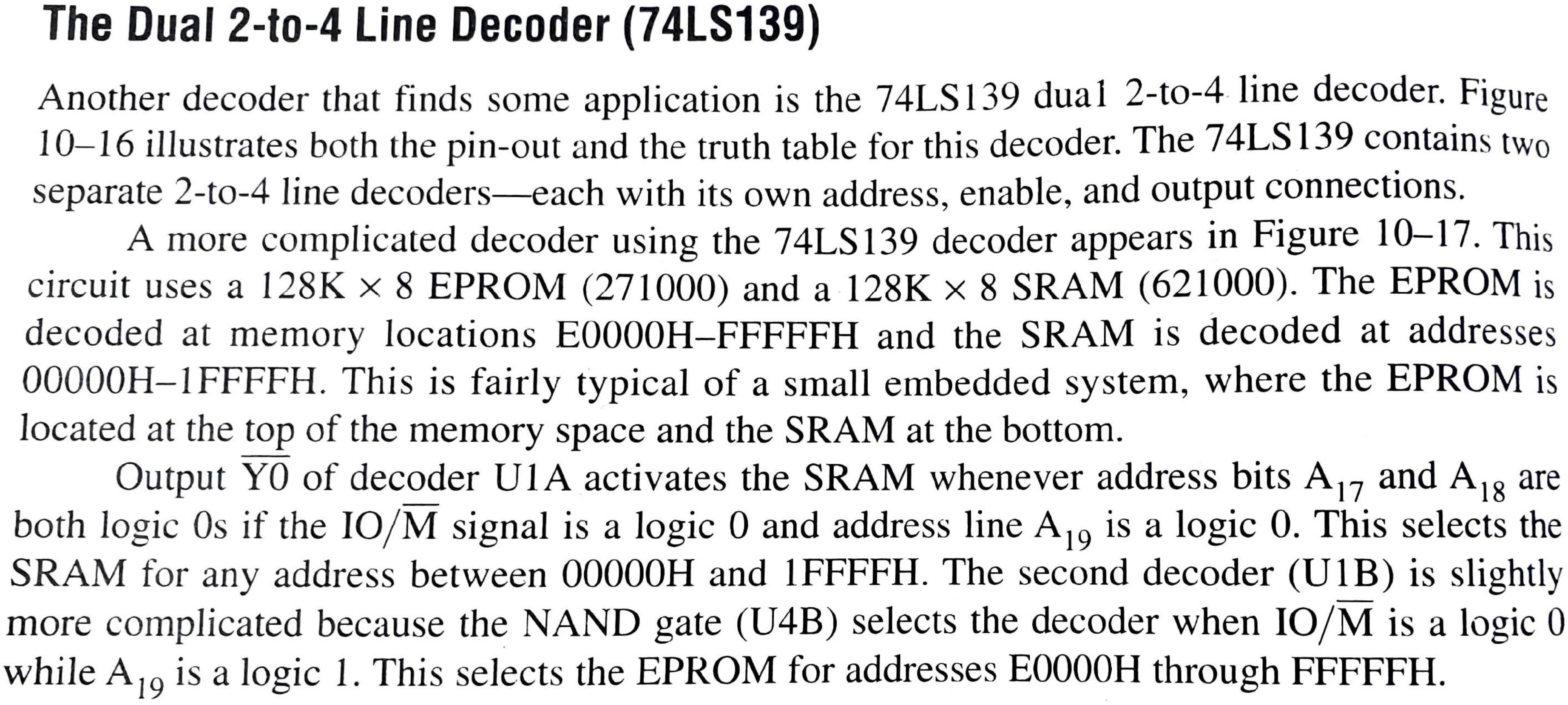


- The EPROM interface uses a 74LS138 (3-to-8 line decoder) plus 8 2732 (4K X 8) EPROMs.
- The EPROM will also require the generation of a wait state.
  - o The EPROM has an access time of 450ns .
  - o The 74LS138 requires 12ns to decode.
  - o The 8086 runs at 5MHz and only allows 460ns for memory to access data.
  - o A wait state adds 200ns of additional time

8086 8-Bit Memory



8-Bit Memory Interface using 2 to 4 Line Decoder



The Dual 2-to-4 Line Decoder (74LS139)

Another decoder that finds some application is the 74LS139 dual 2-to-4 line decoder. Figure 10–16 illustrates both the pin-out and the truth table for this decoder. The 74LS139 contains two separate 2-to-4 line decoders—each with its own address, enable, and output connections.

A more complicated decoder using the 74LS139 decoder appears in Figure 10–17. This circuit uses a 128K × 8 EPROM (271000) and a 128K × 8 SRAM (621000). The EPROM is decoded at memory locations E0000H–FFFFFH and the SRAM is decoded at addresses 00000H–1FFFFH. This is fairly typical of a small embedded system, where the EPROM is located at the top of the memory space and the SRAM at the bottom.

Output $\overline{Y0}$ of decoder U1A activates the SRAM whenever address bits A_{17} and A_{18} are both logic 0s if the IO/$\overline{M}$ signal is a logic 0 and address line A_{19} is a logic 0. This selects the SRAM for any address between 00000H and 1FFFFH. The second decoder (U1B) is slightly more complicated because the NAND gate (U4B) selects the decoder when IO/$\overline{M}$ is a logic 0 while A_{19} is a logic 1. This selects the EPROM for addresses E0000H through FFFFFH.

8-Bit Memory Interface using 2 to 4 Line Decoder

THE INTEL MICROPROCESSORS

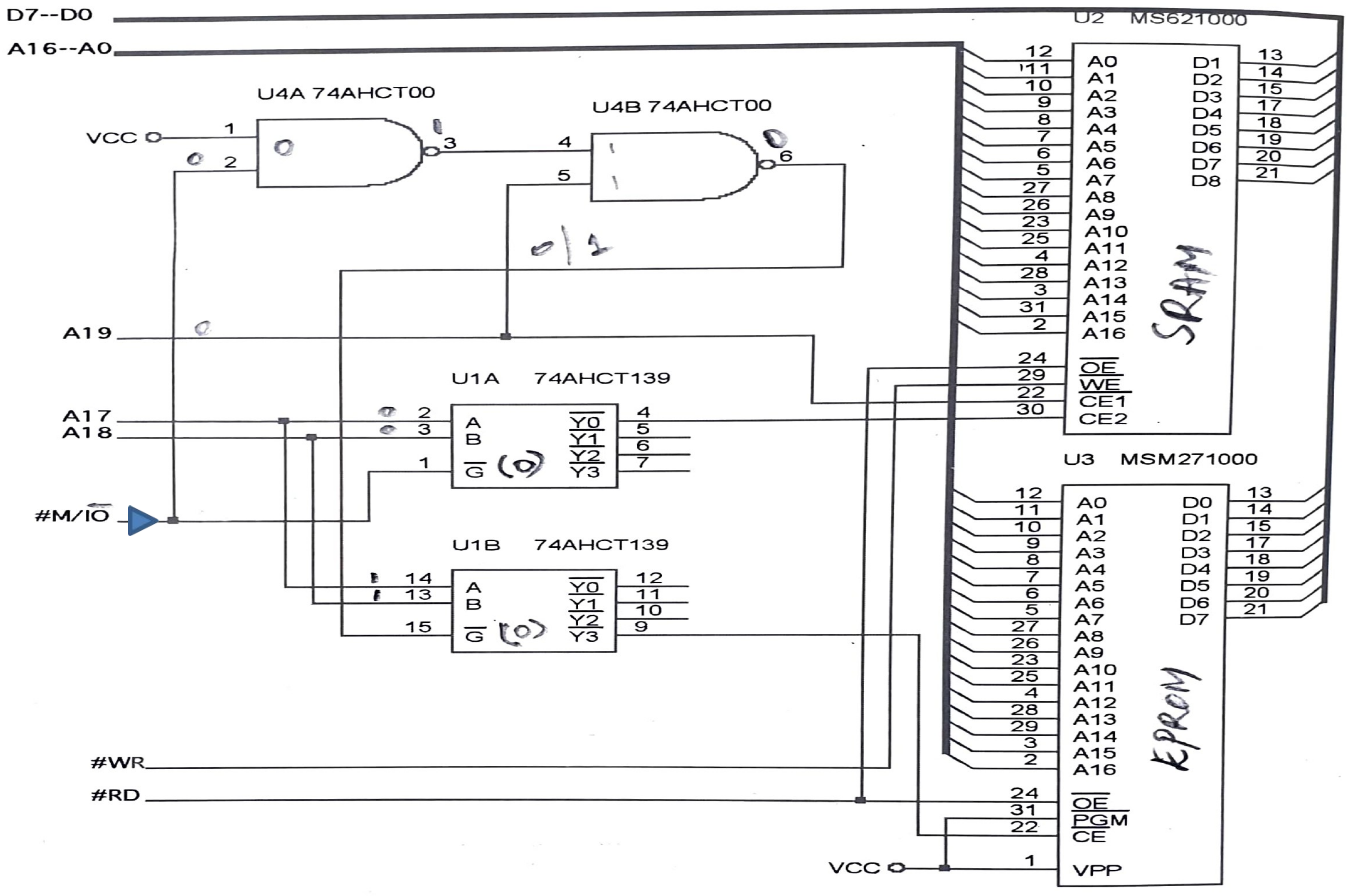


FIGURE 10–17 A sample memory system constructed with a 74HCT139.

Simplified Solution

THE INTEL MICROPROCESSORS

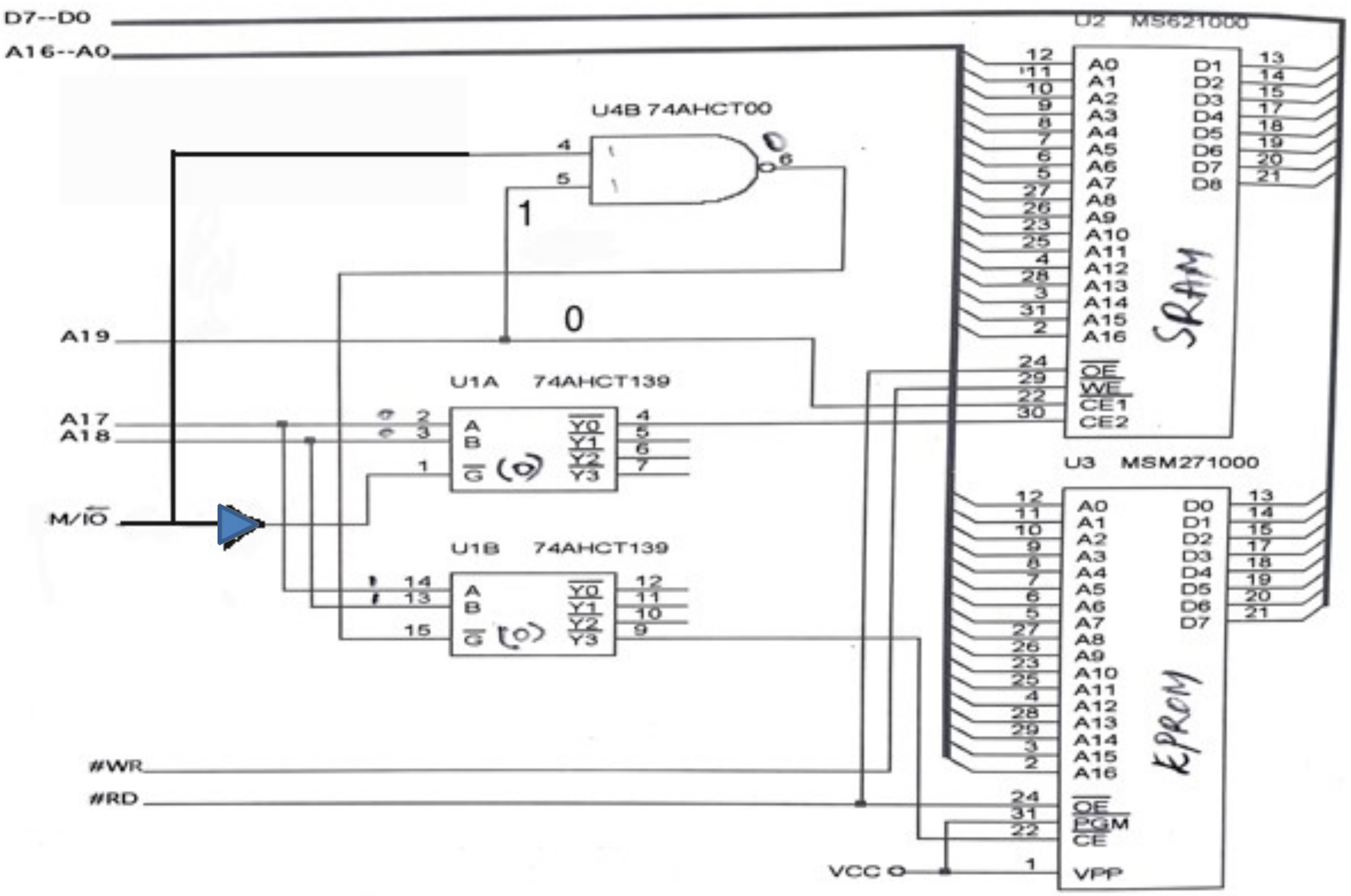
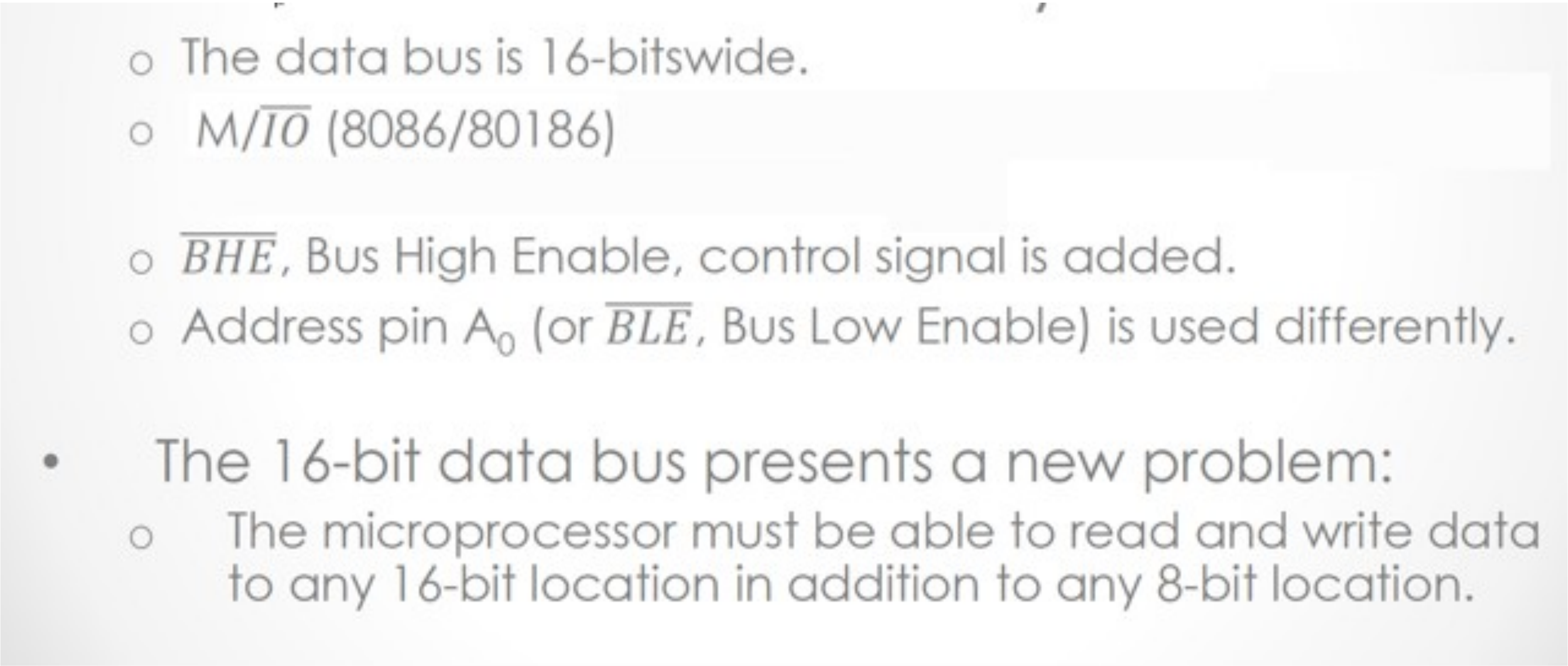


FIGURE 10–17 A sample memory system constructed with a 74HCT139.

Address Range

Input			Chip Address																	Address	
A19	**A18**	**A17**	**A16**	**A15**	**A14**	**A13**	**A12**	**A11**	**A10**	**A9**	**A8**	**A7**	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**	**Range**	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	00000H	**SRAM**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	00001H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	00002H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	00003H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	00004H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	00005H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	00006H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	00007H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	00008H	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	00009H	
0	0	0																		-----	
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1FFFFH	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	E000H	**EPROM**
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	E0001H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	E0002H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	E0003H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	E0004H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	E0005H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	E0006H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	E0007H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	E0008H	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	E0009H	
1	1	1																		-----	
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	FFFFFH	

16-Bit Memory

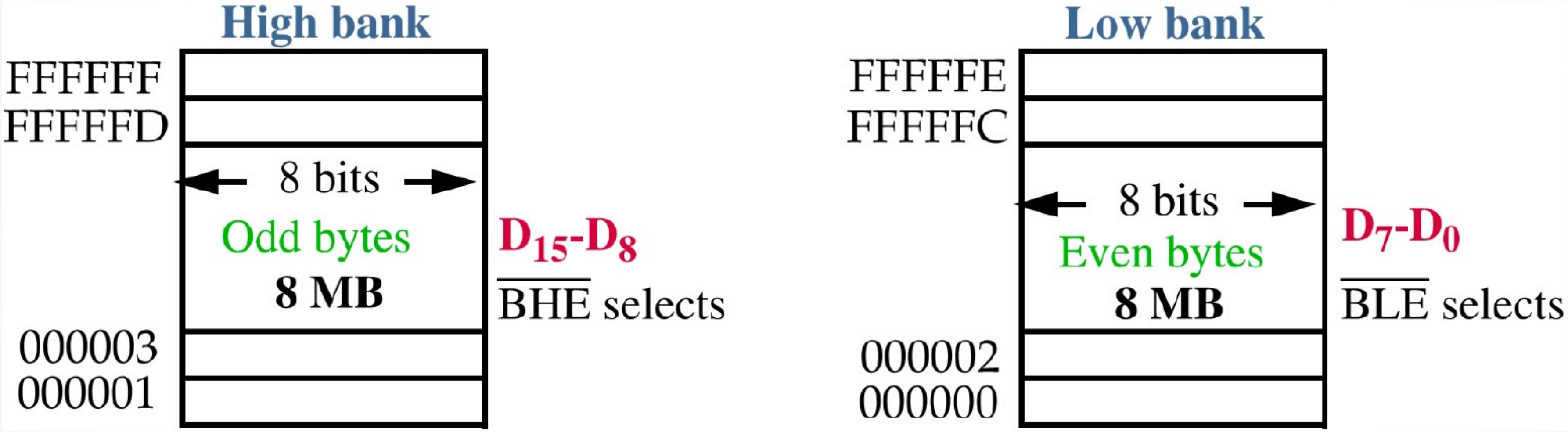


- The data bus is 16-bitswide.
- M/$\overline{IO}$ (8086/80186)
- $\overline{BHE}$, Bus High Enable, control signal is added.
- Address pin A_0 (or $\overline{BLE}$, Bus Low Enable) is used differently.

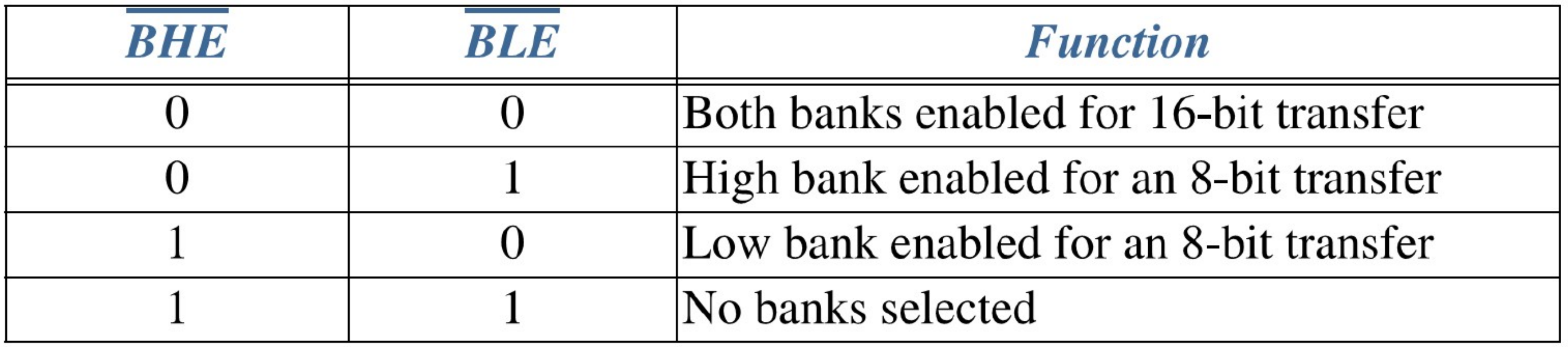
- The 16-bit data bus presents a new problem:
  - The microprocessor must be able to read and write data to any 16-bit location in addition to any 8-bit location.

16-Bit Memory

- The data bus and memory are divided into banks:

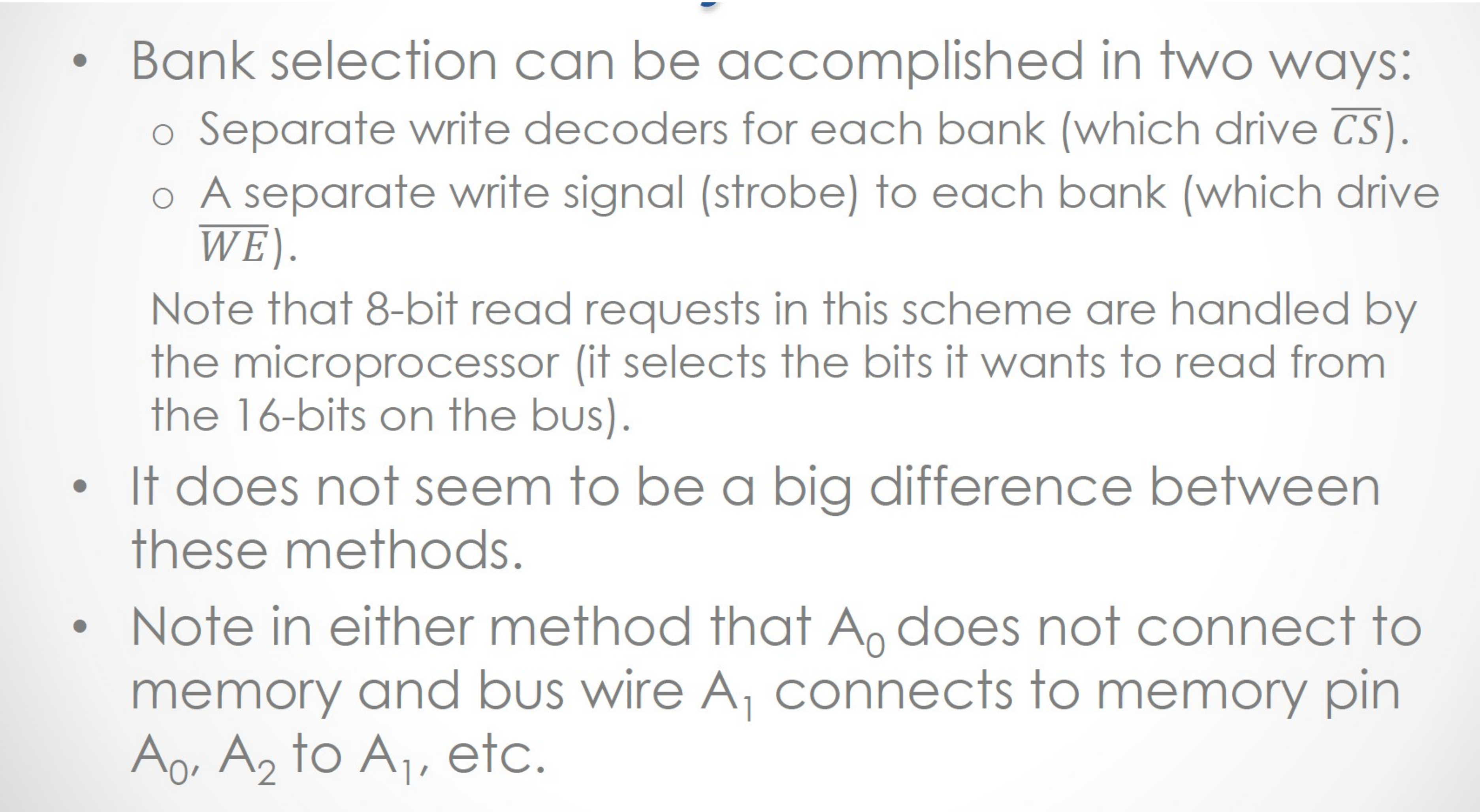


- BHE and BLE are used to select one or both:



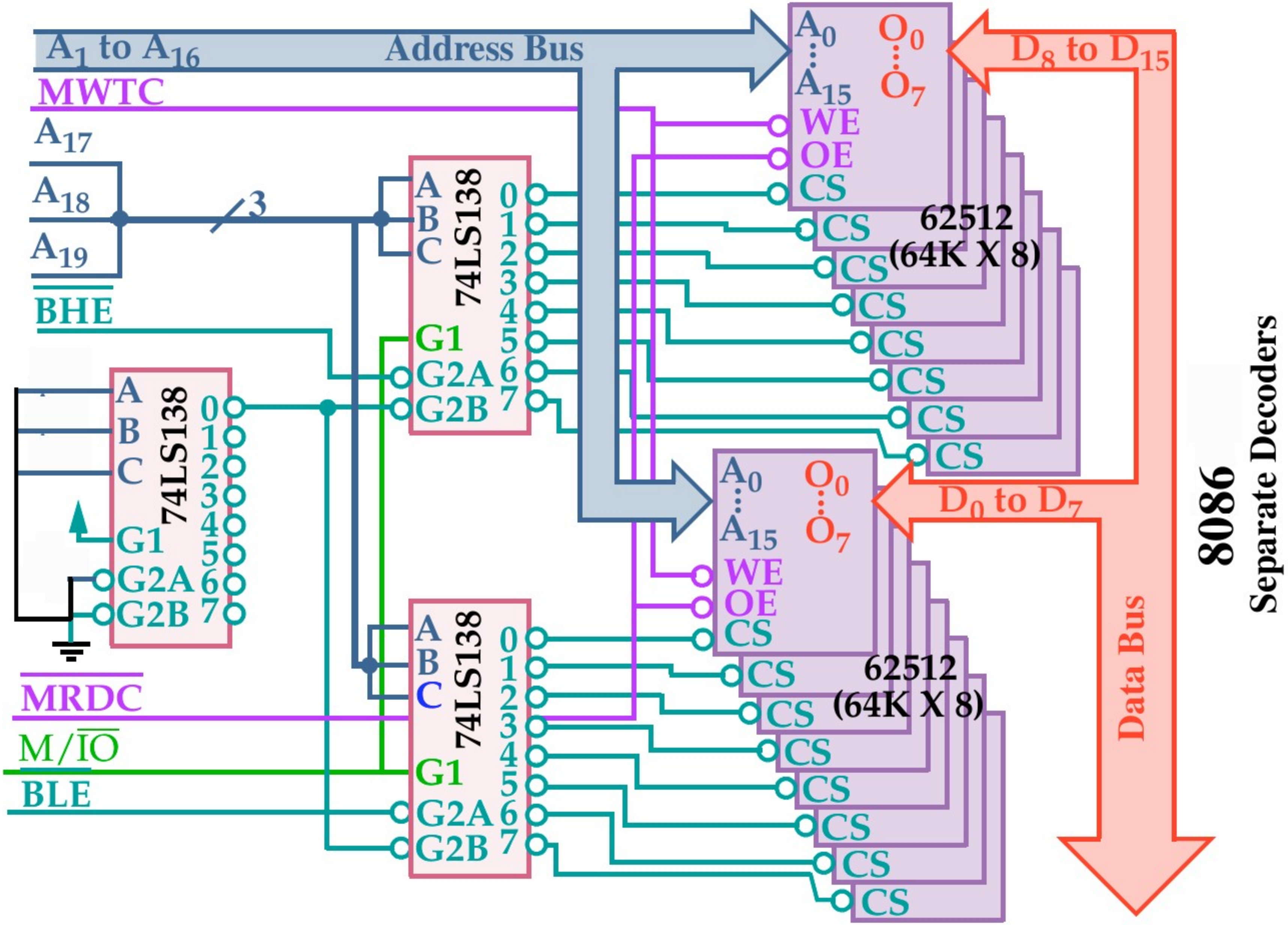
$\overline{BHE}$	$\overline{BLE}$	*Function*
0	0	Both banks enabled for 16-bit transfer
0	1	High bank enabled for an 8-bit transfer
1	0	Low bank enabled for an 8-bit transfer
1	1	No banks selected

16-Bit Memory

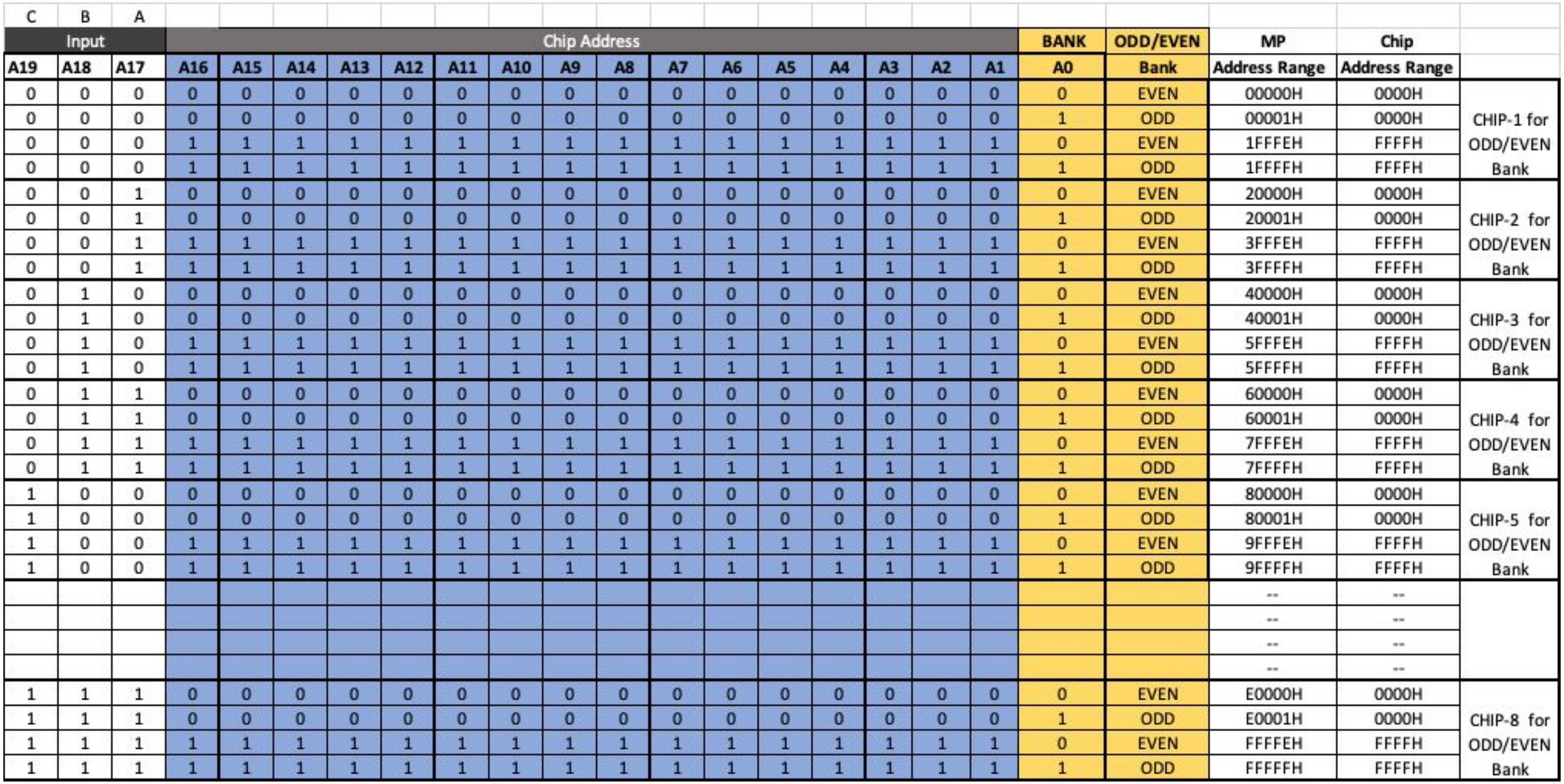


- Bank selection can be accomplished in two ways:
  - Separate write decoders for each bank (which drive $\overline{CS}$).
  - A separate write signal (strobe) to each bank (which drive $\overline{WE}$).

  Note that 8-bit read requests in this scheme are handled by the microprocessor (it selects the bits it wants to read from the 16-bits on the bus).
- It does not seem to be a big difference between these methods.
- Note in either method that A_0 does not connect to memory and bus wire A_1 connects to memory pin A_0, A_2 to A_1, etc.


A1 to A16
Address Bus
MWTC
A17
A18
3
A19
BHE
A
B
C
74LS138
G1
G2A
G2B
0
1
2
3
4
5
6
7
MRDC
M/IO
BLE
A0
A15
O0
O7
WE
OE
CS
62512
(64K X 8)
D8 to D15
D0 to D7
Data Bus
8086
Separate Decoders

Address Range



C	B	A																					
Input			Chip Address																BANK	ODD/EVEN	MP	Chip	
A19	A18	A17	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	Bank	Address Range	Address Range	
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EVEN	00000H	0000H	CHIP-1 for ODD/EVEN Bank
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	ODD	00001H	0000H	
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	EVEN	1FFFEH	FFFFH	
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ODD	1FFFFH	FFFFH	
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EVEN	20000H	0000H	CHIP-2 for ODD/EVEN Bank
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	ODD	20001H	0000H	
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	EVEN	3FFFEH	FFFFH	
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ODD	3FFFFH	FFFFH	
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EVEN	40000H	0000H	CHIP-3 for ODD/EVEN Bank
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	ODD	40001H	0000H	
0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	EVEN	5FFFEH	FFFFH	
0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ODD	5FFFFH	FFFFH	
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EVEN	60000H	0000H	CHIP-4 for ODD/EVEN Bank
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	ODD	60001H	0000H	
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	EVEN	7FFFEH	FFFFH	
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ODD	7FFFFH	FFFFH	
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EVEN	80000H	0000H	CHIP-5 for ODD/EVEN Bank
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	ODD	80001H	0000H	
1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	EVEN	9FFFEH	FFFFH	
1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ODD	9FFFFH	FFFFH	
																					--	--	
																					--	--	
																					--	--	
																					--	--	
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	EVEN	E0000H	0000H	CHIP-8 for ODD/EVEN Bank
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	ODD	E0001H	0000H	
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	EVEN	FFFFEH	FFFFH	
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ODD	FFFFFH	FFFFH	

Reference

- Brey, B.B., 2009. *The Intel microprocessors: 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro processor, Pentium II, Pentium III, Pentium 4, and Core2 with 64-bit extensions: architecture, programming, and interfacing*. Pearson Education India.