Lecture 19-20: March 26, 2021

Computer Architecture and Organization-I

Biplab K Sikdar

Example of memory interfacing

Assume CPU has - 16 address lines (A_{15}, A_{14}, A_{13}, $\cdots$, A_1, A_0), 8-bit bidirectional data bus (D), $RD/\overline{WR}$ and $IO/\overline{M}$ to define CPU access: I/O device or memory.

$IO/\overline{M}$ = 1 $\Rightarrow$ access to input/output device.

$IO/\overline{M}$ = 0 $\Rightarrow$ access to memory.

This CPU provides 2^{16} = 64K memory address space (0000_h to $FFFF_h$).

It implements I/O mapped I/O - separate address space for memory ($IO/\overline{M}$ = 0).

In some earlier processors, the scheme memory mapped I/O was implemented.

Interface a memory (RAM/ROM) chip of 16K$\times$8.

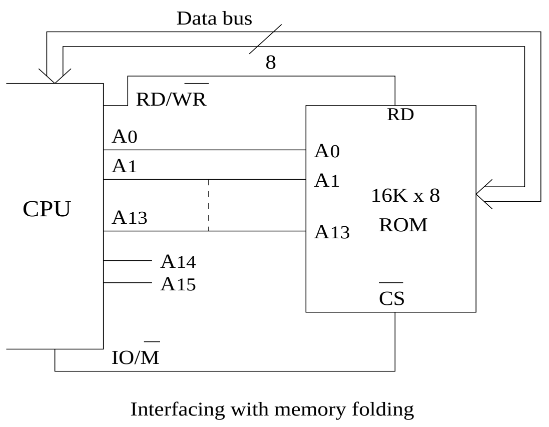

Figure 9: Memory interfacing option 1

Simplest form of interfacing logic is shown in Figure 9.

16K ROM has 14 address lines (A_{13} to A_0). $IO/\overline{M}$ is connected to CS of ROM.

Address space occupied by ROM chip can be

		A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	$\cdots$	A_1	A_0	
1.	*Address of LS location*	0	0	0	0	0	$\cdots$	0	0	0000_h
	Address of MS location	0	0	1	1	1	$\cdots$	1	1	$3FFF_h$
		A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	$\cdots$	A_1	A_0	
2.	*Address of LS location*	0	1	0	0	0	$\cdots$	0	0	4000_h
	Address of MS location	0	1	1	1	1	$\cdots$	1	1	$7FFF_h$
		A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	$\cdots$	A_1	A_0	
3.	*Address of LS location*	1	0	0	0	0	$\cdots$	0	0	8000_h
	Address of MS location	1	0	1	1	1	$\cdots$	1	1	$BFFF_h$
		A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	$\cdots$	A_1	A_0	
4.	*Address of LS location*	1	1	0	0	0	$\cdots$	0	0	$C000_h$
	Address of MS location	1	1	1	1	1	$\cdots$	1	1	$FFFF_h$

A physical location of memory module can have 4 addresses.

Ex: 4 addesses 0000$\cdots$00, 0100$\cdots$00, 1000$\cdots$00, and 1100$\cdots$00 point to same physical location. This is memory folding.

Avoiding memory folding

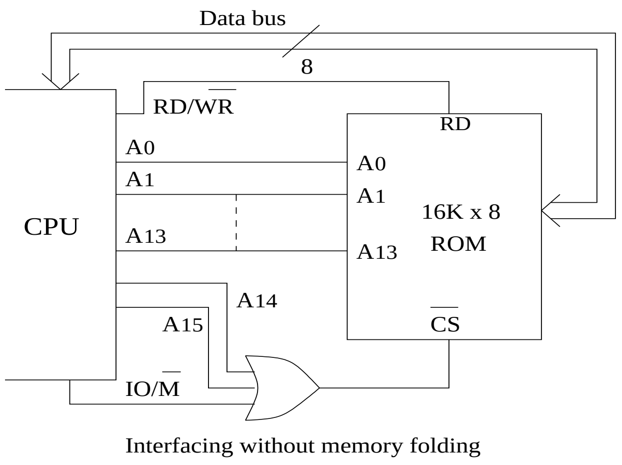

Figure 10: Memory interfacing option 2

Interfacing of Figure 10 avoids memory folding.

Address space occupied by ROM chip is

$$\begin{array}{lccccccccl} & A_{15} & A_{14} & A_{13} & A_{12} & A_{11} & \cdots & A_1 & A_0 & \\ Address\ of\ LS\ location & 0 & 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0000h \\ Address\ of\ MS\ location & 0 & 0 & 1 & 1 & 1 & \cdots & 1 & 1 & 3FFFh \end{array}$$

This design eliminates provision of any further expansion.

Complete decoding

Effective interfacing technique is complete decoding (Figure 11).

Interfacing logic partitions whole memory address space in small slots.

1 of 4 decoder partitions 64K address space in 4 slots each of 16K.

In Figure 11, only one slot is utilized, the other three are for expansion.

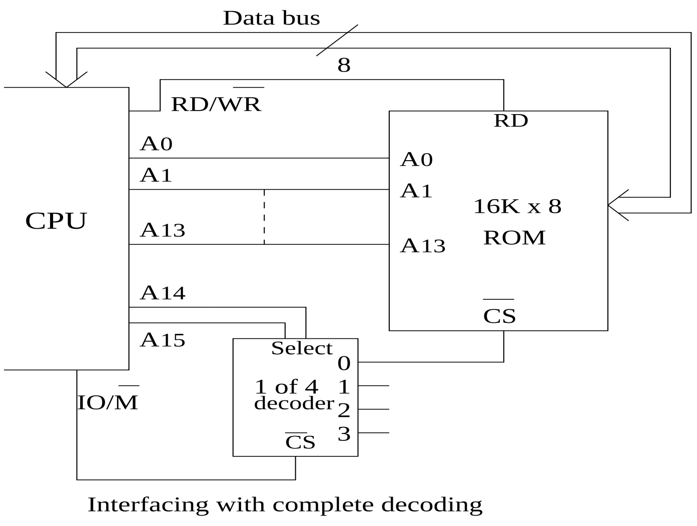

Figure 11: Memory interfacing: complete decoding

Address space occupied by ROM chip is same as Figure 10.

	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	$\cdots$	A_1	A_0	
Address of LS location	0	0	0	0	0	$\cdots$	0	0	$0000h$
Address of MS location	0	0	1	1	1	$\cdots$	1	1	$3FFFh$

Expansion in complete decoding

Figure 12 describes expansion of 8K on Figure 11.

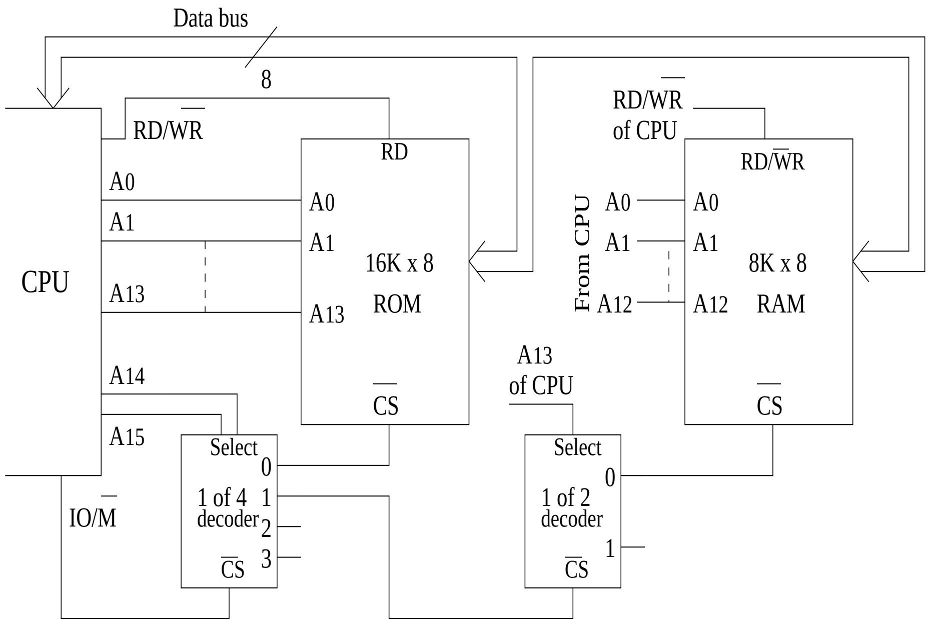

Figure 12: Memory expansion I

Address space occupied by the memory chips

Address of	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	$\cdots$	A_1	A_0		*Chip*
LS location	0	0	0	0	0	$\cdots$	0	0	$0000h$	*ROM*
MS location	0	0	1	1	1	$\cdots$	1	1	$3FFFh$	
LS location	0	1	0	0	0	$\cdots$	0	0	$4000h$	*RAM*
MS location	0	1	0	1	1	$\cdots$	1	1	$5FFFh$	

-that is, ROM occupied from 0000h to 3FFFh and RAM 4000h to 5FFFh.

Expansion in complete decoding

Figure 13 describes expansion of 32K on Figure 11.

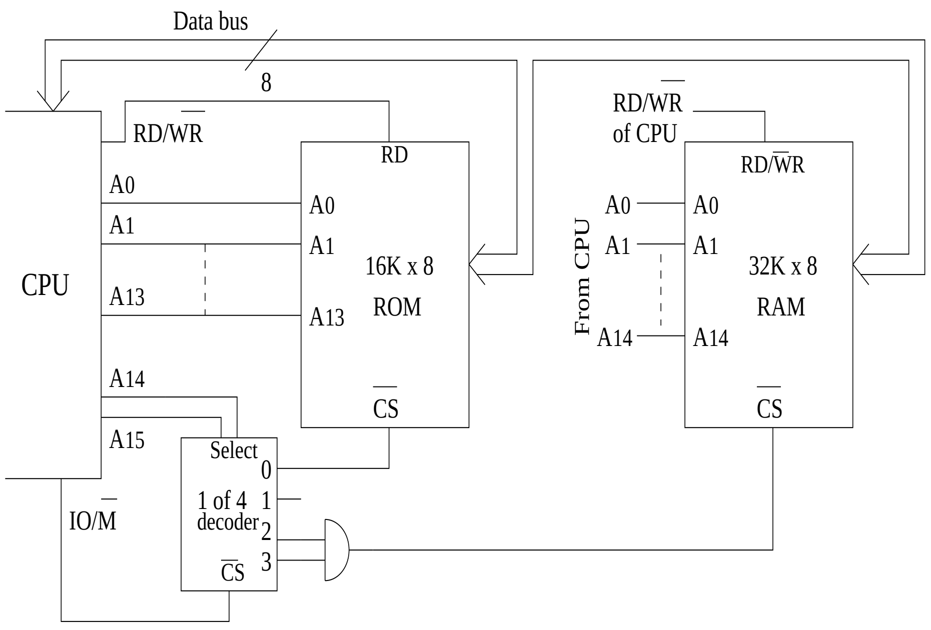

Figure 13: Memory expansion II

Address space occupied by the memory chips

Address of	A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	$\cdots$	A_1	A_0		*Chip*
LS location	0	0	0	0	0	$\cdots$	0	0	$0000h$	*ROM*
MS location	0	0	1	1	1	$\cdots$	1	1	$3FFFh$	
LS location	1	0	0	0	0	$\cdots$	0	0	$8000h$	*RAM*
MS location	1	1	1	1	1	$\cdots$	1	1	$FFFFh$	

-that is, ROM occupied from 0000h to 3FFFh and RAM 8000h to FFFFh.

0.2 High Speed Memories

Memory access time is a bottleneck.

A solution addresses optimal memory reference only through load and store instructions (implemented in RISC). Other solutions target

1. Decrease memory access time by using a faster but expensive technology.
2. Insert small faster memory (cache) in between conventional MM and CPU.
3. Access more than one word from memory in a cycle.
4. Interleave access to more than one word to achieve faster access.
5. Consider large memory word to retrieve more information in one cycle.

0.3 Main Memory Technology

1-D memory Each cell is connected to one address driver. If storage capacity is N bits, access circuitry needs one N-output decoder and N address drivers (Figure 14).

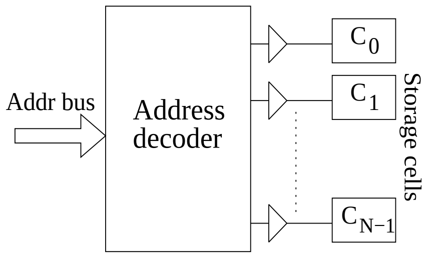

Figure 14: 1-dimensional memory module

2-D memory In 2-dimensional memory module, number of cells =

$2^{k_1} \times 2^{k_2}$, where $k_1 + k_2 = m$.

Number of address drivers is = $2^{k_1} + 2^{k_2}$ (Figure 15).

This can also be modified as

2^{k_1} rows $\times$ 2^{m+k_2} columns (2^m columns are selected by each one of 2^{k_2} columns).

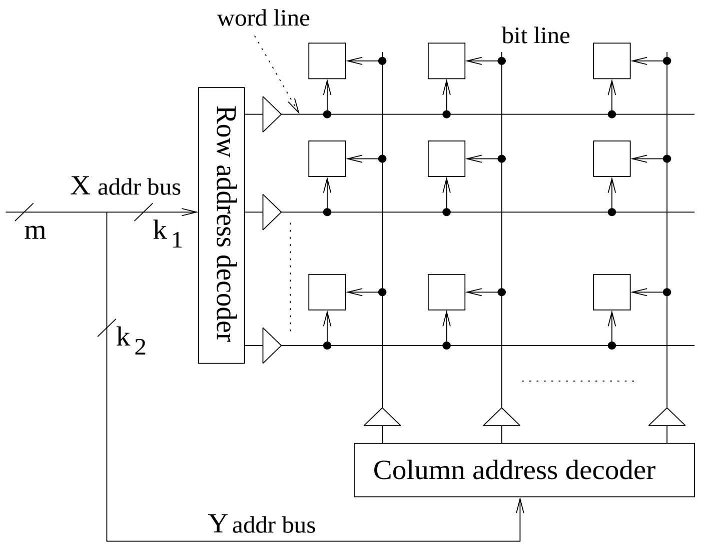

Figure 15: 2-dimensional memory module

3-D memory Home task.

0.3.1 Constructing large memory module

Objective is to solve memory design problem that a computer architect considers:

Given $2^m \times d$-bit RAM modules/chips how to design an $2^{m_1} \times d_1$-bit RAM module, where $m_1 \geq m$ and $d_1 \geq d$.

Memory with 2^m words each of d bits is called $2^m \times d$-bit memory.

General approach - construct $2^p \times q$ array of $M_{2^m,d}$, where p=$\lceil m_1/m \rceil$ and q=$\lceil d_1/d \rceil$.

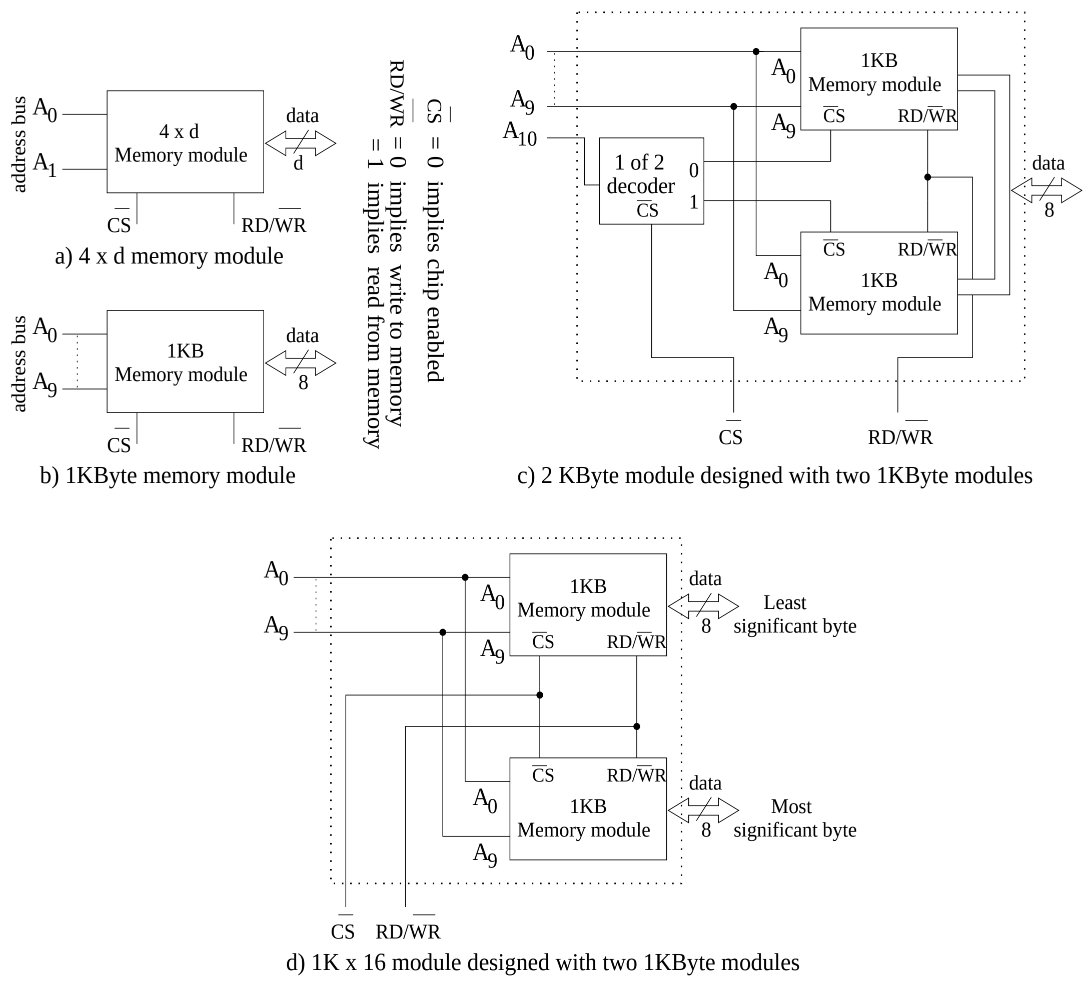

Figure 16: Memory arrays

<u>QUIZ</u>