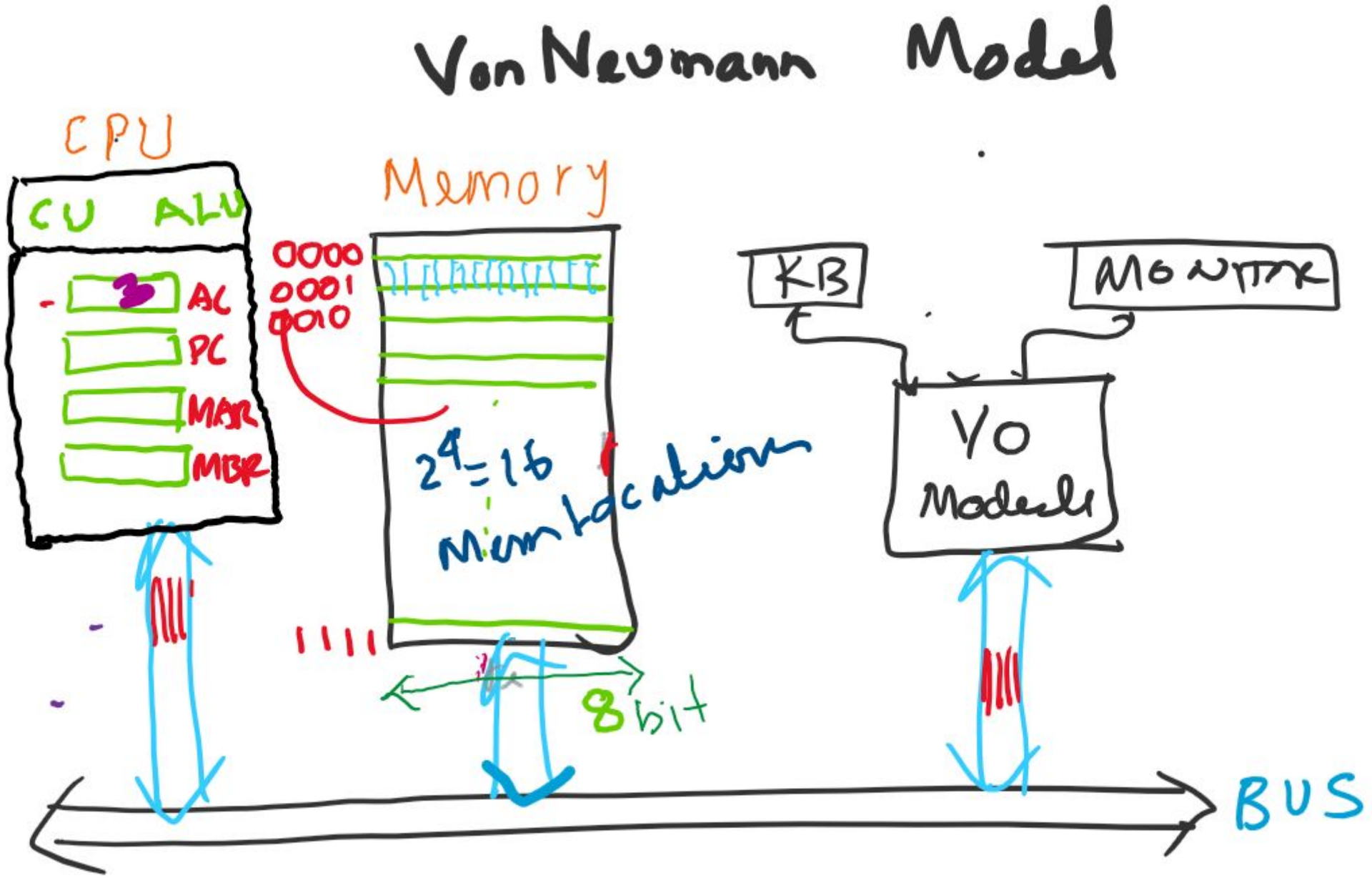
FIT9137
Introduction to Computer Architecture and Networks

Week 3: Memory Design & Management

Safi Uddin

www.shutterstock.com • 548025055

A 4-bit Memory Address & 16 Memory Locations

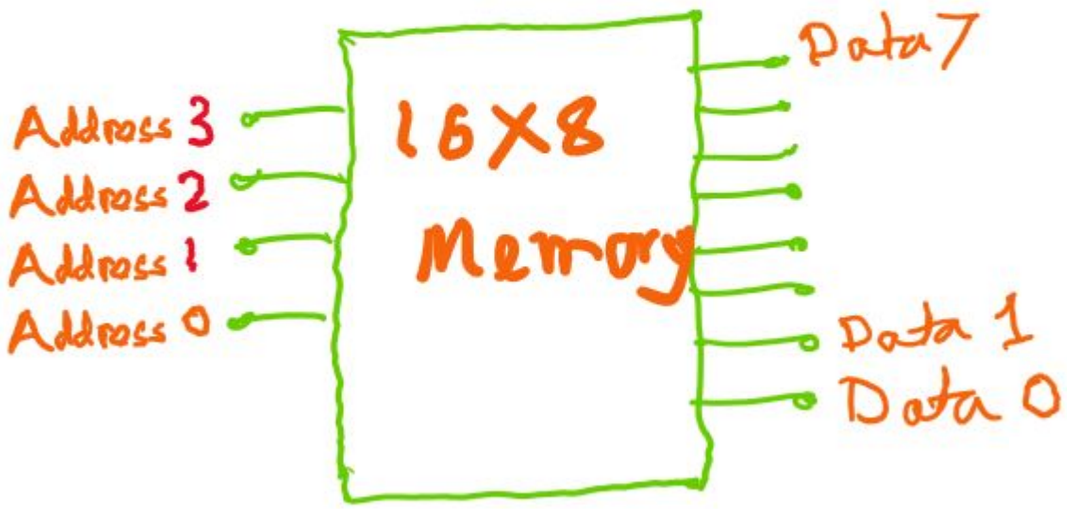


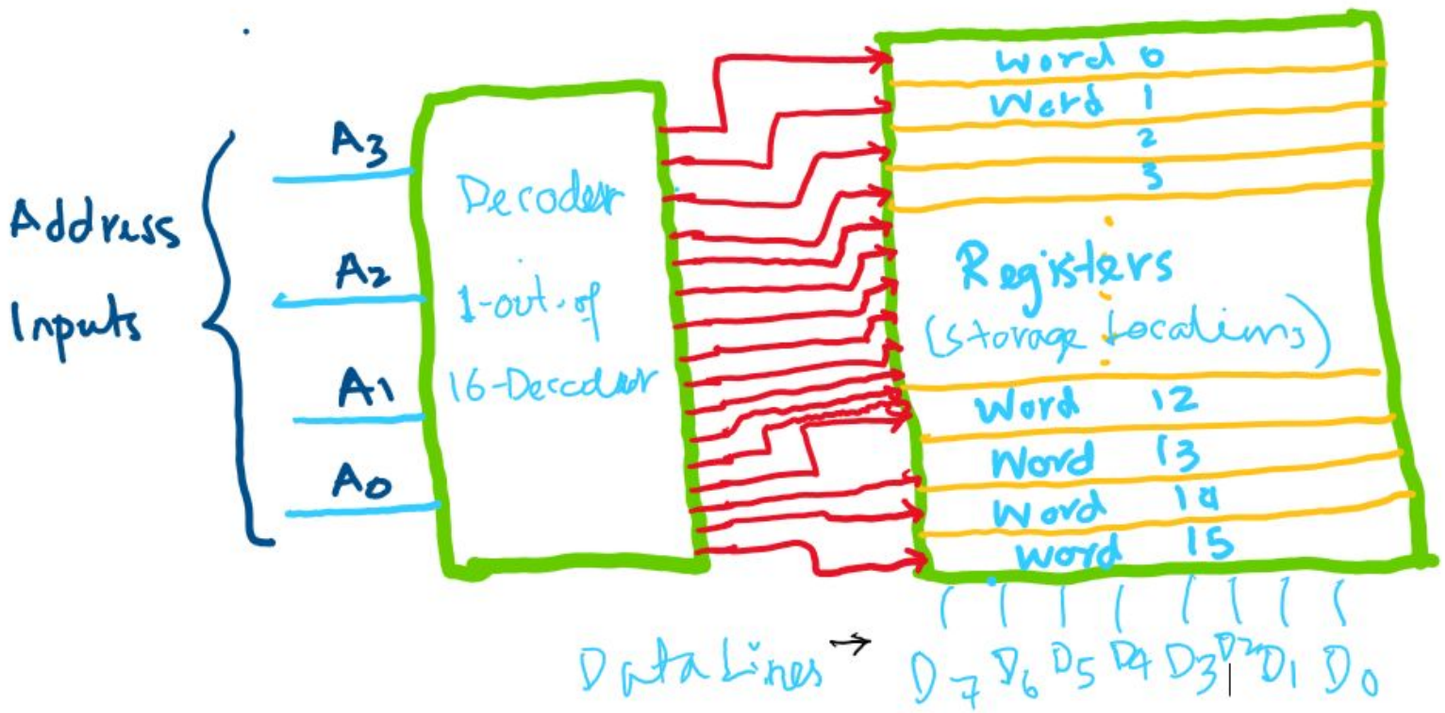
4-bit Memory Address & 16 Locations

	Address				Data							
Word	A_3	A_2	A_1	A_0	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	1	1	0	1	1	1	1	0
1	0	0	0	1	0	0	1	1	1	0	1	0
2	0	0	1	0	1	0	0	0	0	1	0	1
3	0	0	1	1	1	0	1	0	1	1	1	1
4	0	1	0	0	0	0	0	1	1	0	0	1
5	0	1	0	1	0	1	1	1	1	0	1	1
6	0	1	1	0	0	0	0	0	0	0	0	0
7	0	1	1	1	1	1	1	0	1	1	0	1
8	1	0	0	0	0	0	1	1	1	1	0	0
9	1	0	0	1	1	1	1	1	1	1	1	1
10	1	0	1	0	1	0	1	1	1	0	0	0
11	1	0	1	1	1	1	0	0	0	1	1	1
12	1	1	0	0	0	0	1	0	0	1	1	1
13	1	1	0	1	0	1	1	0	1	0	1	0
14	1	1	1	0	1	1	0	1	0	0	1	0
15	1	1	1	1	0	1	0	1	1	0	1	1

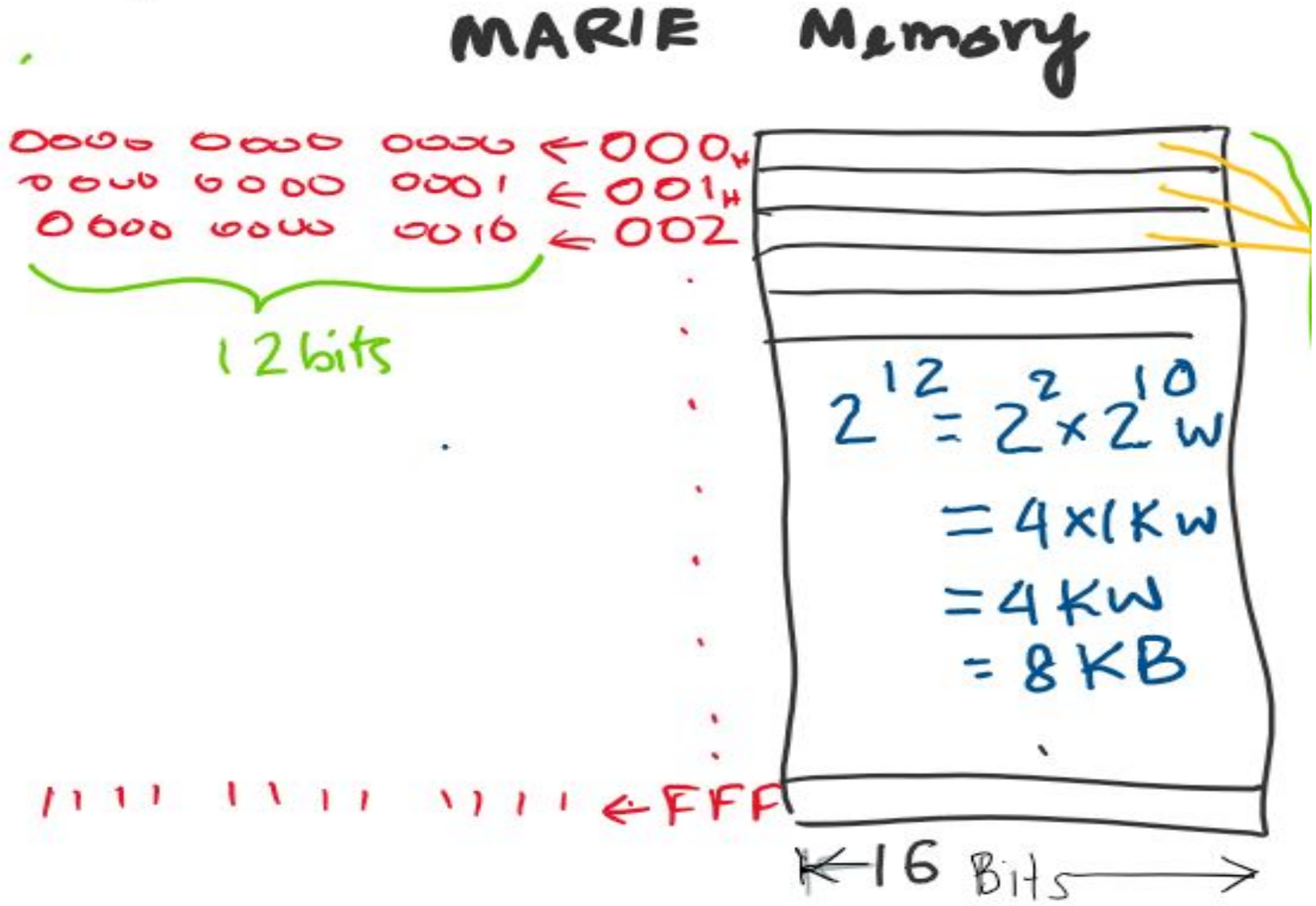
4-bit Memory Address & 16 Locations with A Decoder

A 16X8 Memory

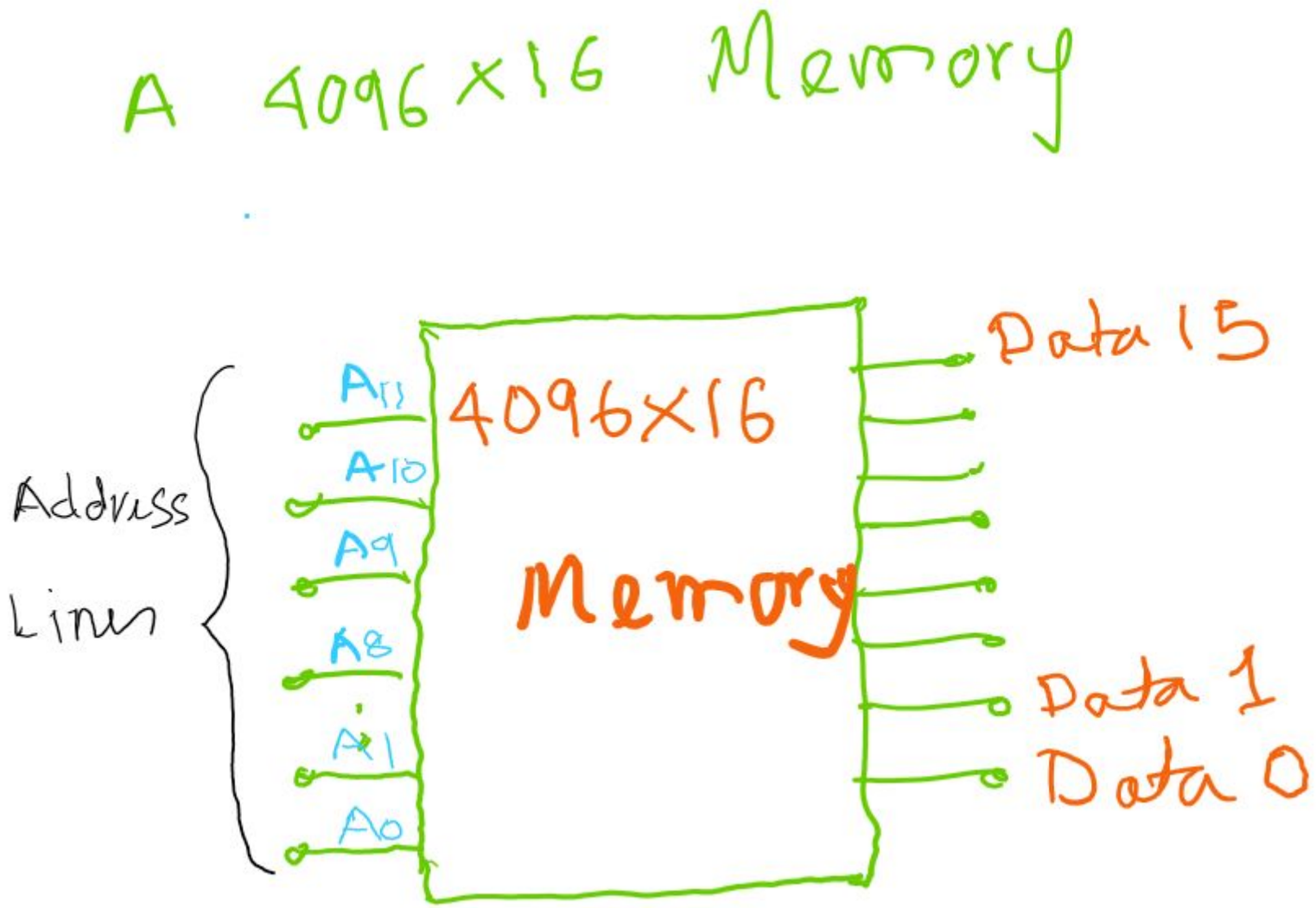




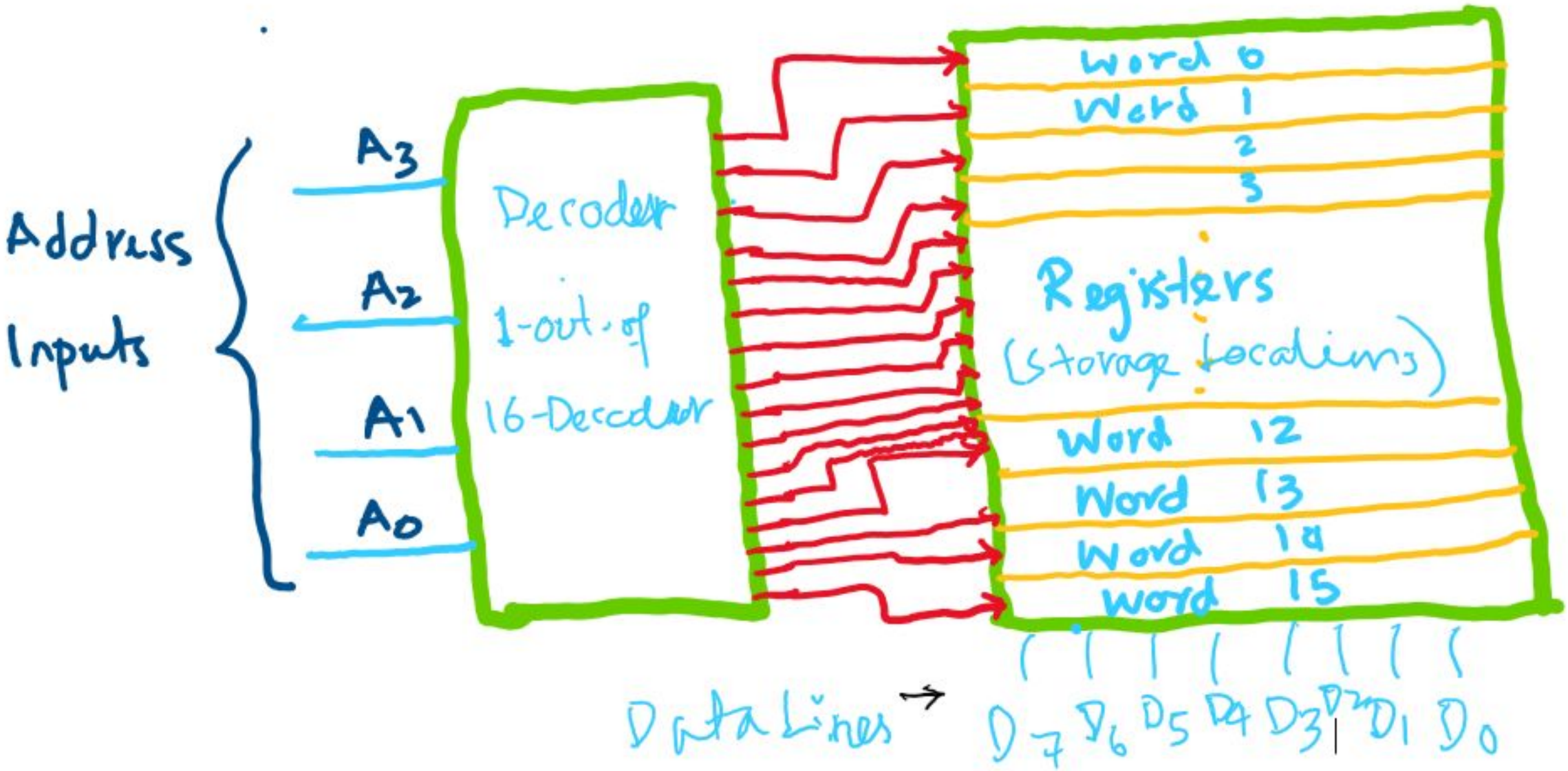
12-bit Memory Address & 4096 Locations



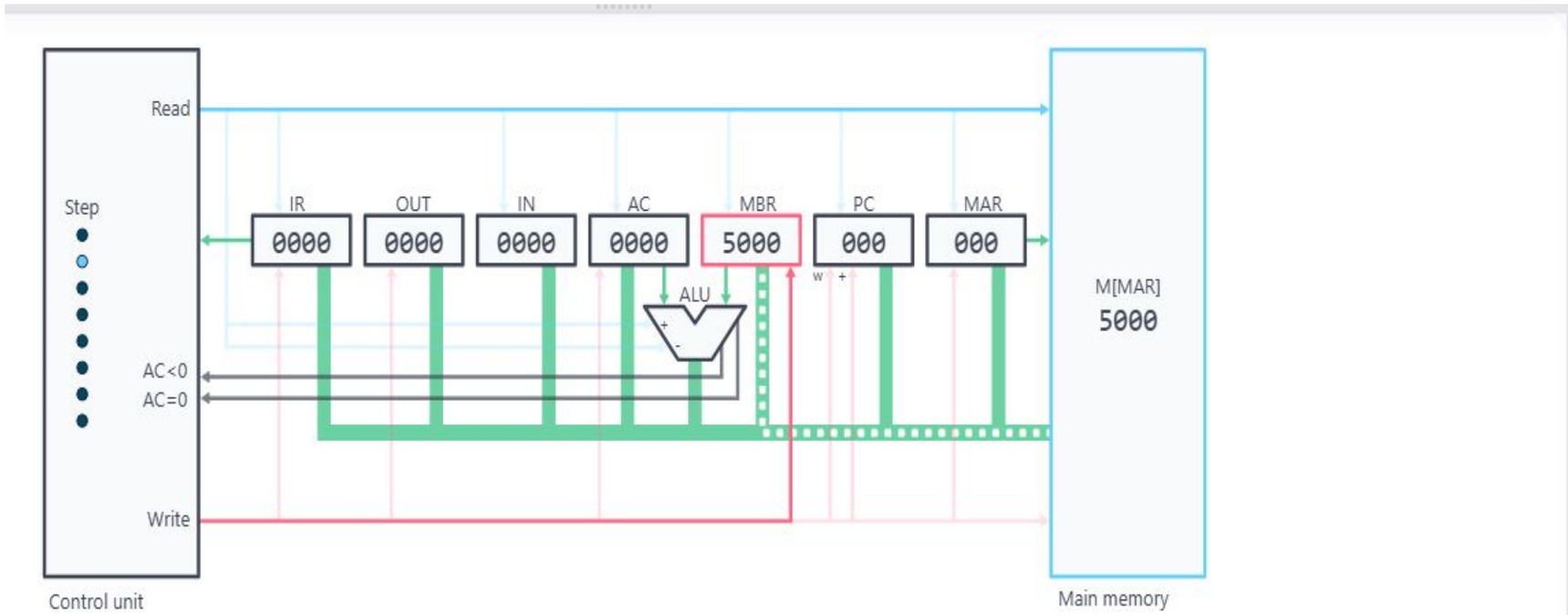
12-bit Memory Address & 4096 Locations



Selecting A Memory Location Using Decoder



Selecting A Memory Using Decoder



RTL log

MAR ← PC
MBR ← M[MAR]

Watch list

Inputs

Inputs Format

Enter input

Hexadecimal

+ Add additional input

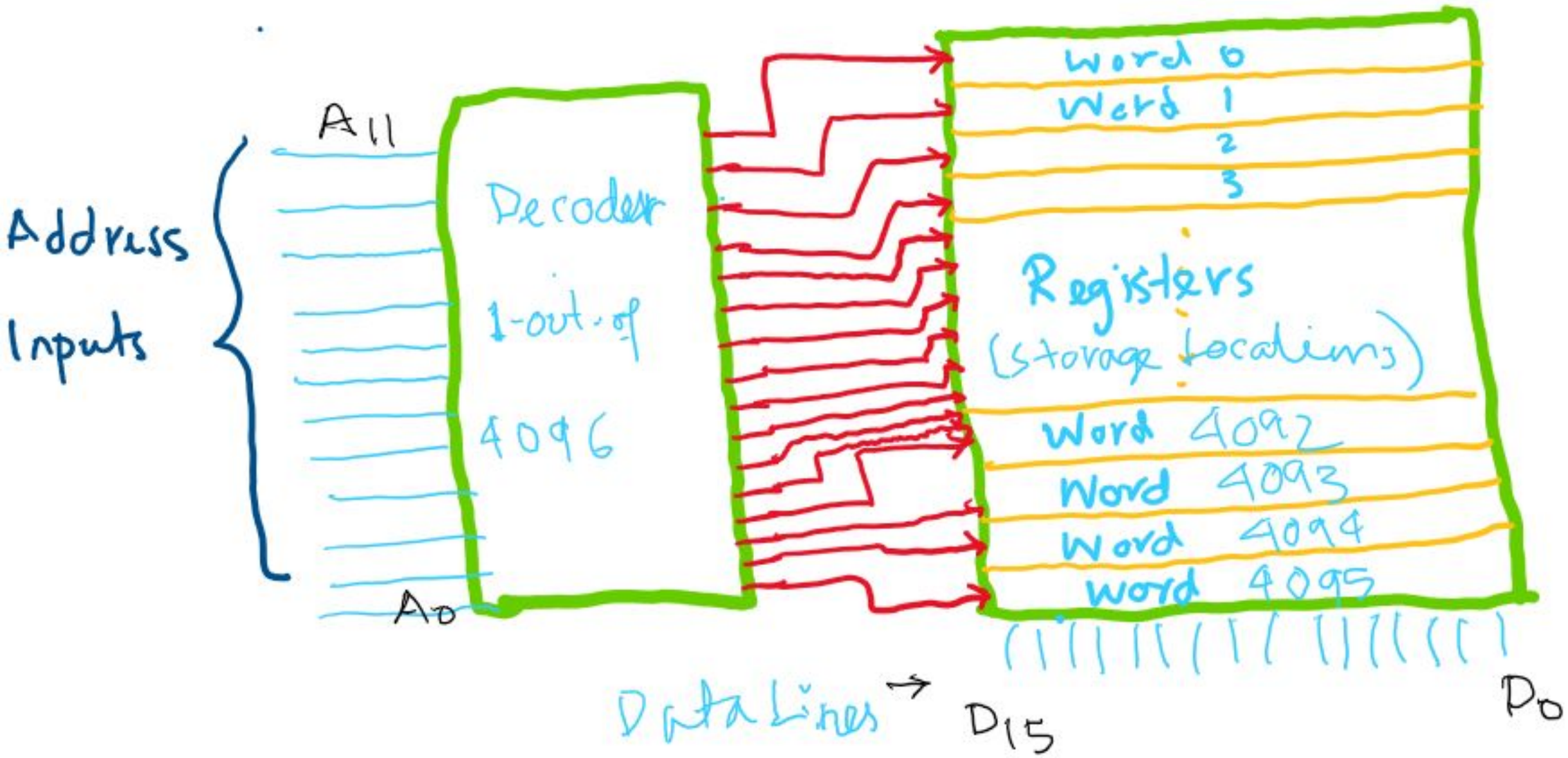
Display

Run ▶ | Step | Micro step | Restart | Speed:

C 0000 IR 0000 MAR 000 MBR 5000 PC 000 IN 0000 OUT 0000

+2	+3	+4	+5	+6	+7	+8	+9	+A	+B	+C	+D	+E	+F
0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000

Selecting A Memory Using Decoder



Activity A:

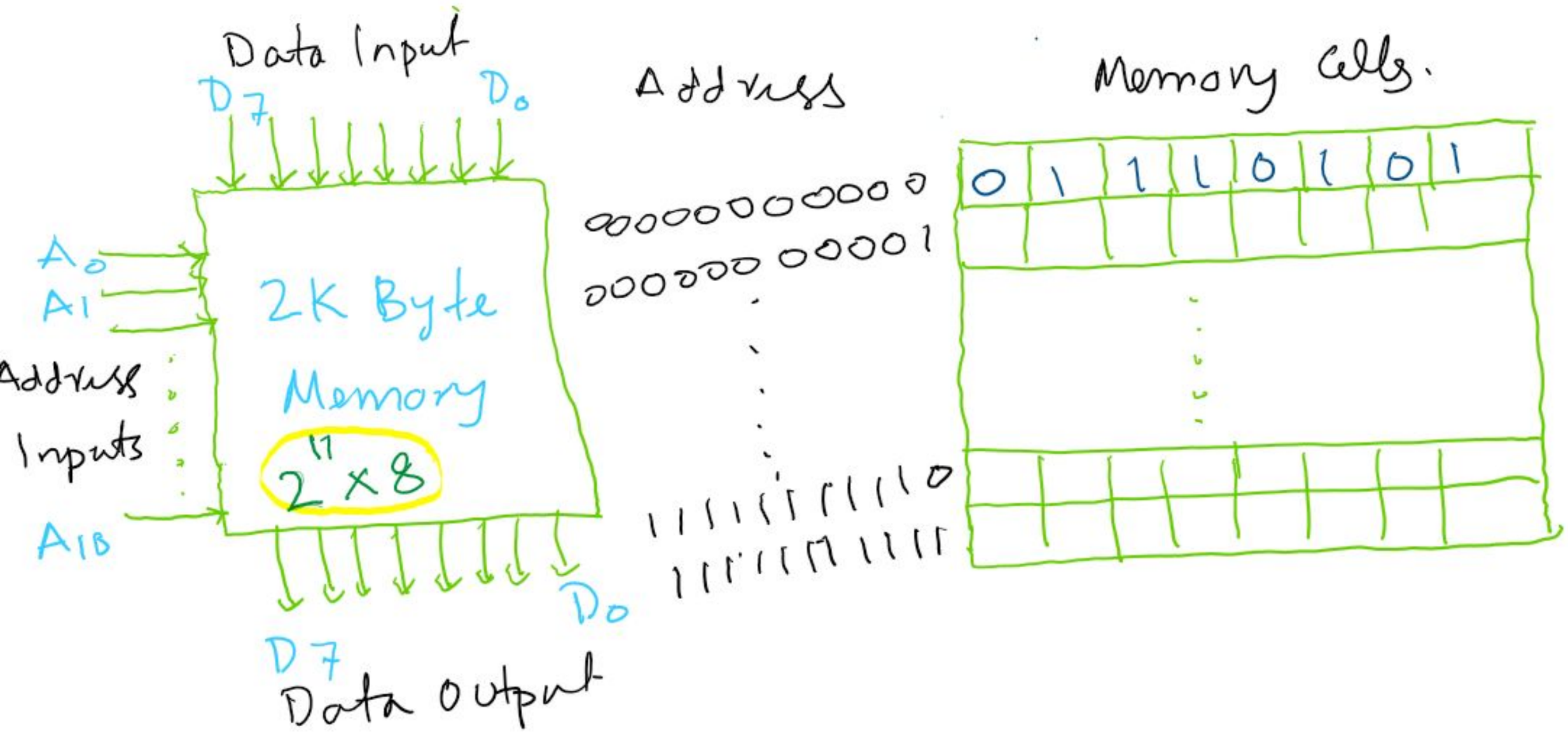
Random Access Memory (RAM) Design using Memory Chips

RAM design using Memory Chips

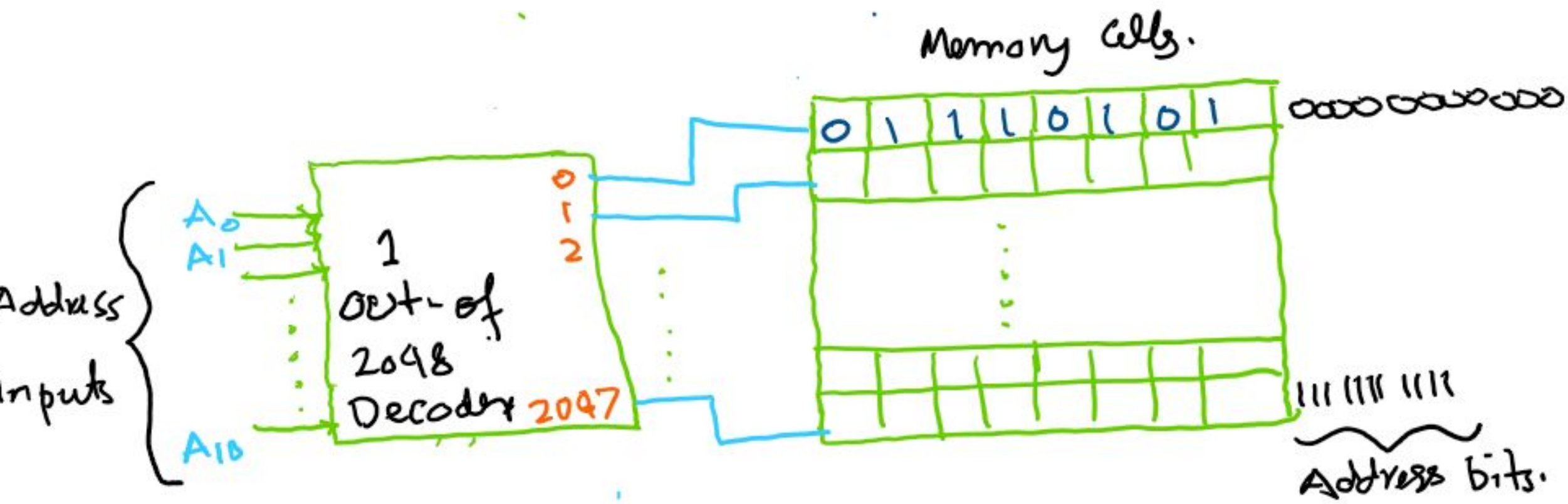
Design a 1-Mbit Random Access Memory (RAM) using 2K Byte (2^{11}X8) Memory Chip.

1. 2K Byte (2^{11} X 8) Memory Chip ? How many address bits (lines)?
2. 2K Byte (2^{11} X 8) Memory Chip ? How many data bits to be read /write when we select these address bits (lines)?
3. A schematic diagram?

RAM design using Memory Chips



RAM design using Memory Chips



RAM design using Memory Chips

Design a 1-Mbit Random Access Memory (RAM) using 2K Byte (2^{11}X8) Memory Chip.

1. 2K Byte (2^{11} X 8) Memory Chip ? 11 address bits (lines)
2. 2K Byte (2^{11} X 8) Memory Chip ? 8 data bits to be read /write when we select 11 address bits (lines).
3. **The size of the memory to be designed: 1M bit = 2^7 X 1K Byte**
4. How many 2KB chips we need to build this new RAM?

RAM design using Memory Chips

Design a 1-Mbit Random Access Memory (RAM) using 2K Byte (2^{11}X8) Memory Chip.

1. 2K Byte (2^{11} X 8) Memory Chip ? 11 address bits (lines)
2. 2K Byte (2^{11} X 8) Memory Chip ? 8 data bits to be read /write when we select 11 address bits (lines).
3. The size of the memory to be designed: 1M bit = 2^7 X 1K Byte
4. **How many 2KB chips we need to build this new RAM?**
   - **= $(2^7$ X 2^{10} Byte)/(2 X 2^{10} X Byte)**
   - **= 2^6**
   - **= 64**

RAM design using Memory Chips

Design a 1-Mbit Random Access Memory (RAM) using 2K Byte (2^{11}X8) Memory Chip.

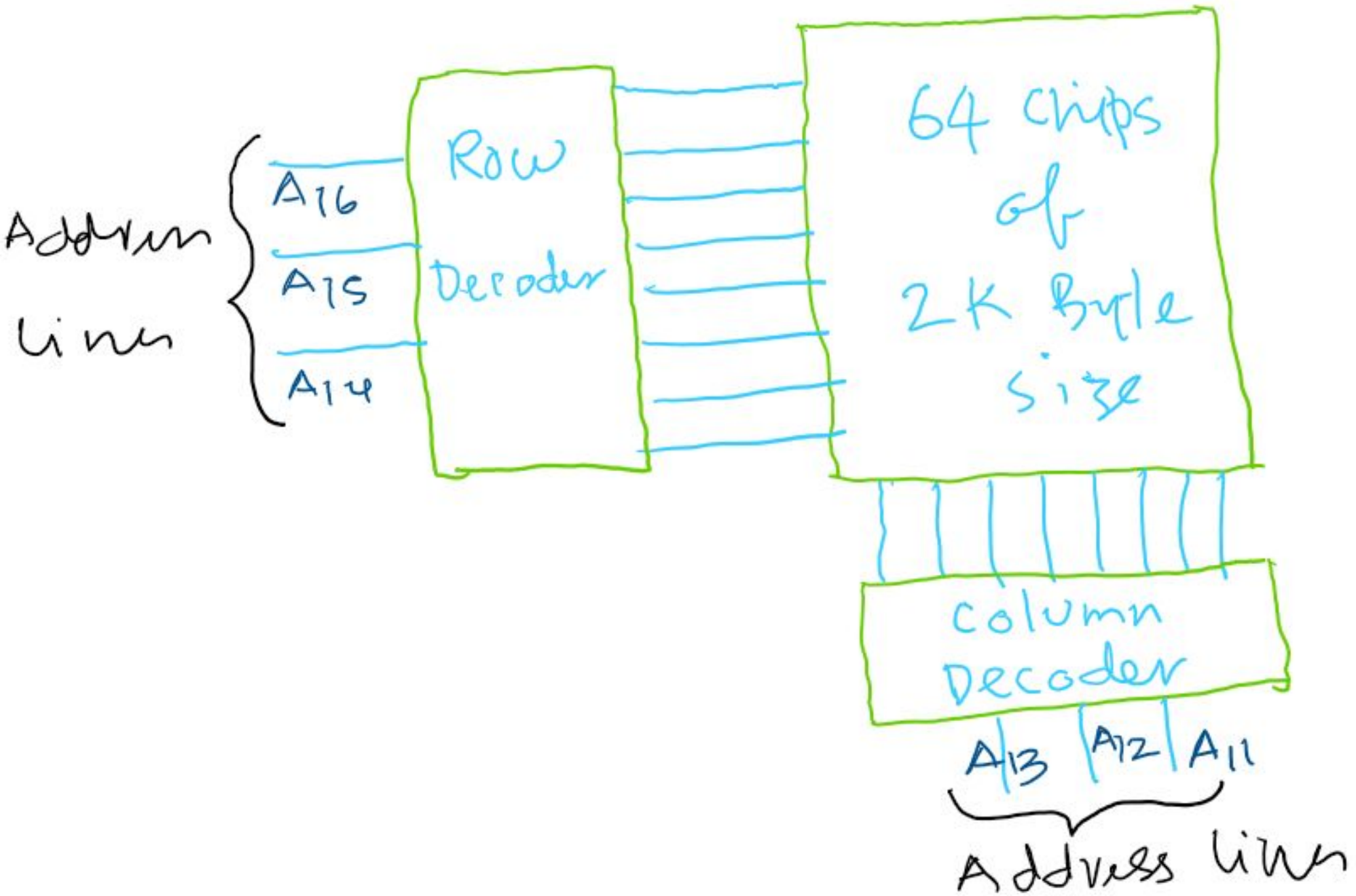
1. 2K Byte (2^{11} X 8) Memory Chip ? 11 address bits (lines)
2. 2K Byte (2^{11} X 8) Memory Chip ? 8 data bits to be read /write when we select 11 address bits (lines).
3. The size of the memory to be designed: 1M bit = 2^7 X 1K Byte
4. How many 2KB chips we need to build this new RAM?
   - **= (2^7 X 2^{10} Byte)/(2 X 2^{10} X Byte) = 2^6 = 64**
5. **We can arrange this 64 chips of size 2K Byte in a grid arrangement.**

RAM design using Memory Chips

RAM design using Memory Chips

- 8 rows and 8 columns in the design
- To select a particular byte in the above memory, we need to select the chip first (1 out of these 64 chips) by mentioning:
  - a row and a column numbers
  - then mention the 11-bit address to select the appropriate byte in that chip.

RAM design using Memory Chips

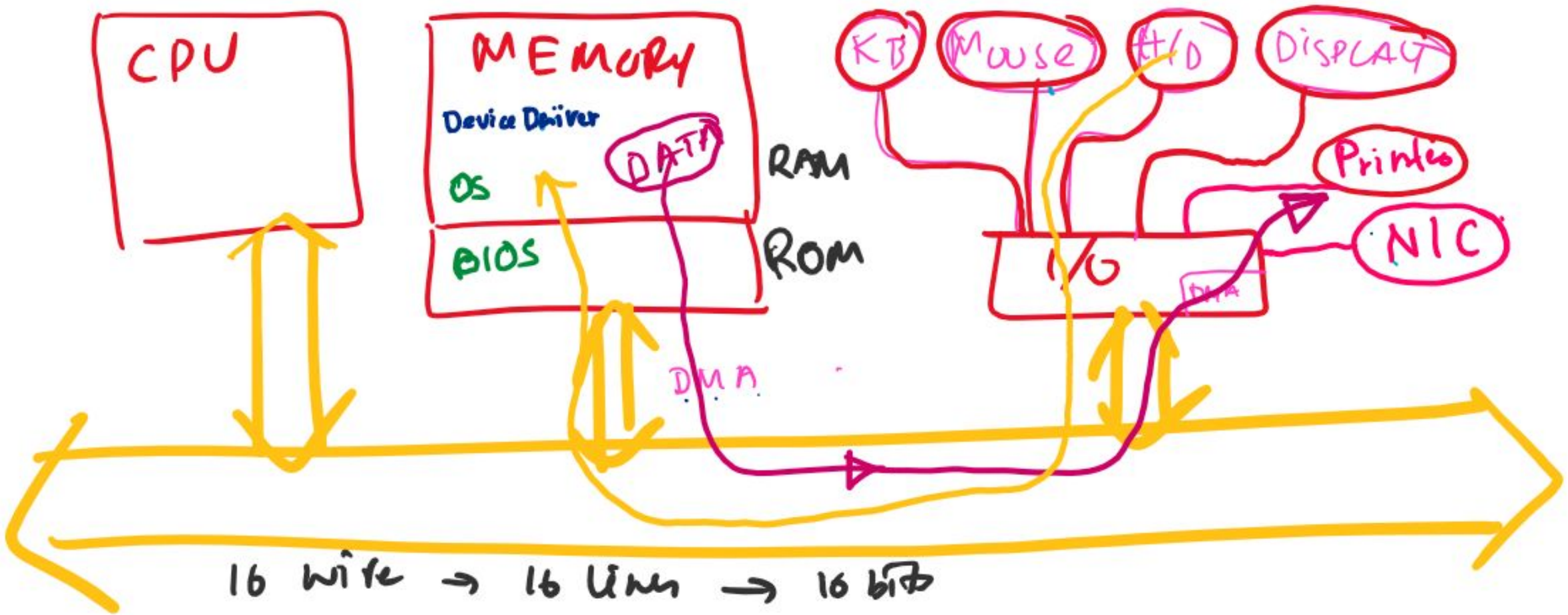


RAM design using Memory Chips

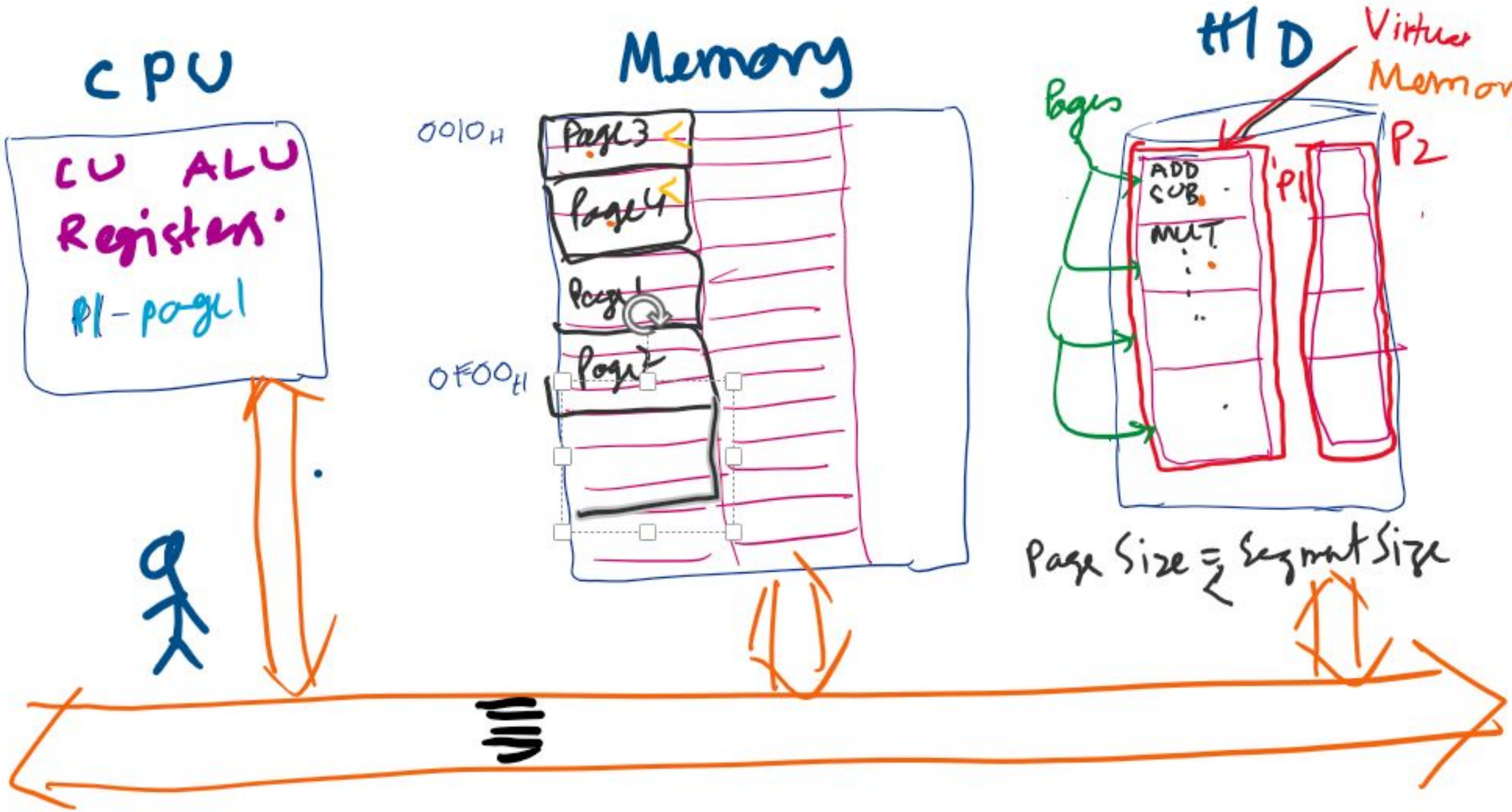
Activity B:

Memory Management - Paging & Page Replacement Algorithms

Who manages the Memory? OS
The Operating System



Paging & Page Replacement Algorithms

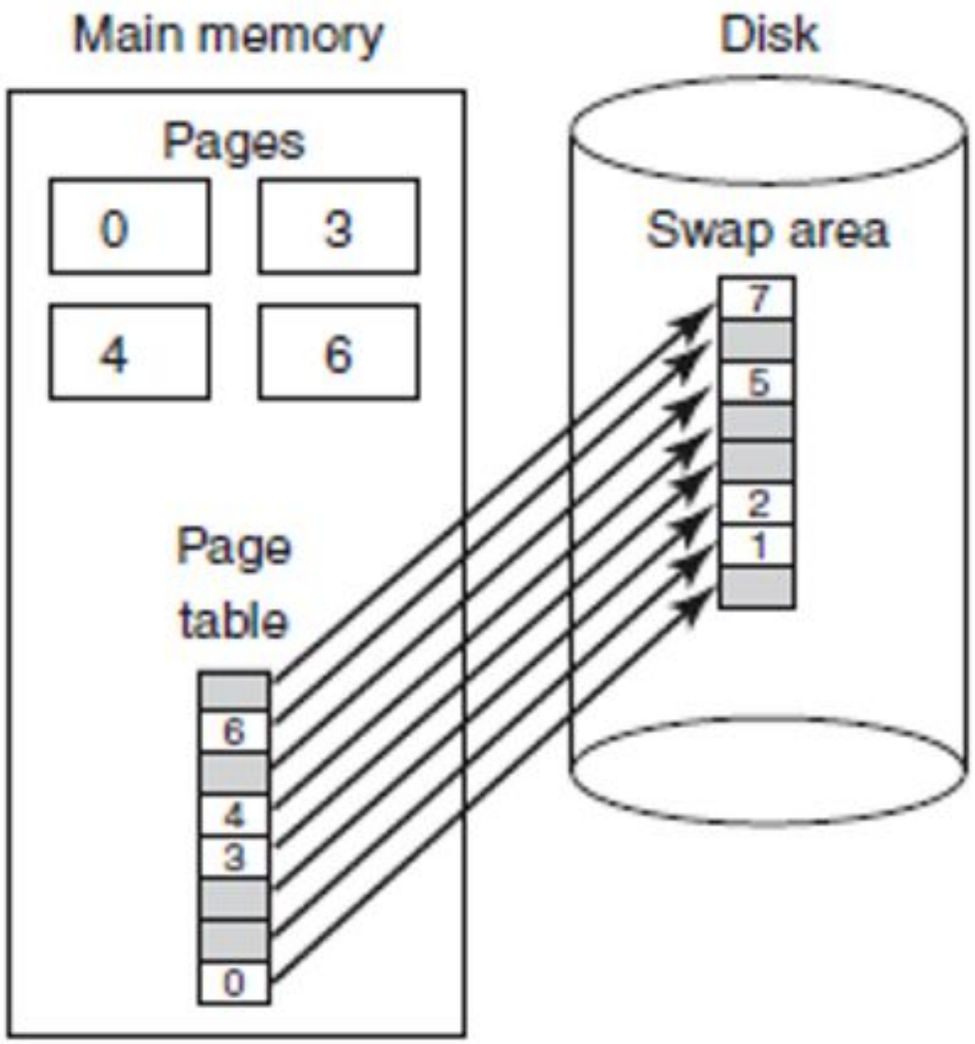


Paging & Page Replacement Algorithms

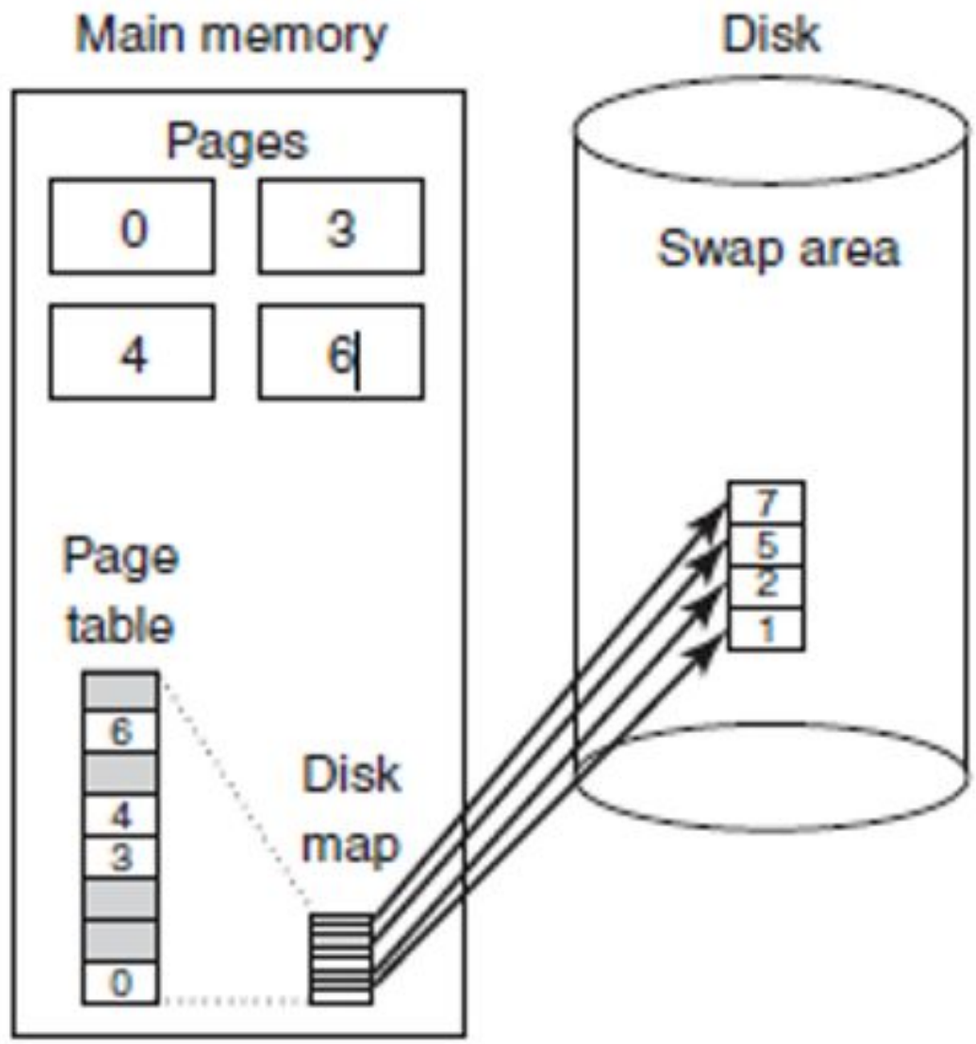
- **Paging:**
  - Programs/data is divided into logical sections called *PAGES* and
  - physical memory is divided into areas called page *FRAMES*.
  - A ***Page-Table*** then translates the pages to their corresponding frames at run-time.

- **Page size and Frame size are equal.**
  - When a program is running, its pages are brought into memory as required – i.e. only the portion being executed is in memory.
  - The rest stays on the hard drive. So we can now run programs which may be larger than the available physical memory.

Paging & Page Replacement Algorithms



(a)



(b)

Paging & Page Replacement Algorithms

Sequence of page requests

Page -> Request	3	2	1	0	3	2	4	3	2	1	0	4
Frame1->												
Frame2->												
Frame3->												

Physical Memory
To store 3 pages
in 3 frames.

Follow the Workshop Acitivties Steps & Answer the Questions

End of the Workshop Tasks

Have a nice week ahead.

See You all Next Week