Digital Electronics and Computer Organization

Digital Design

Lecture 25: Memory

Birla Institute of Technology & Science, Pilani
Hyderabad Campus

Types of memory used in digital systems

RAM (Random Access Memory)

-Can perform both Read and Write Operations

-Stored information is lost when power is turned off

ROM (Read Only Memory)

-Can perform only Read operations

-Suitable information already stored and can be retrieved at any time

-Binary information "Programmed" in to ROM by embedded hardware

Hard Disks

Read / Write Non-volatile

Magnetic SSD

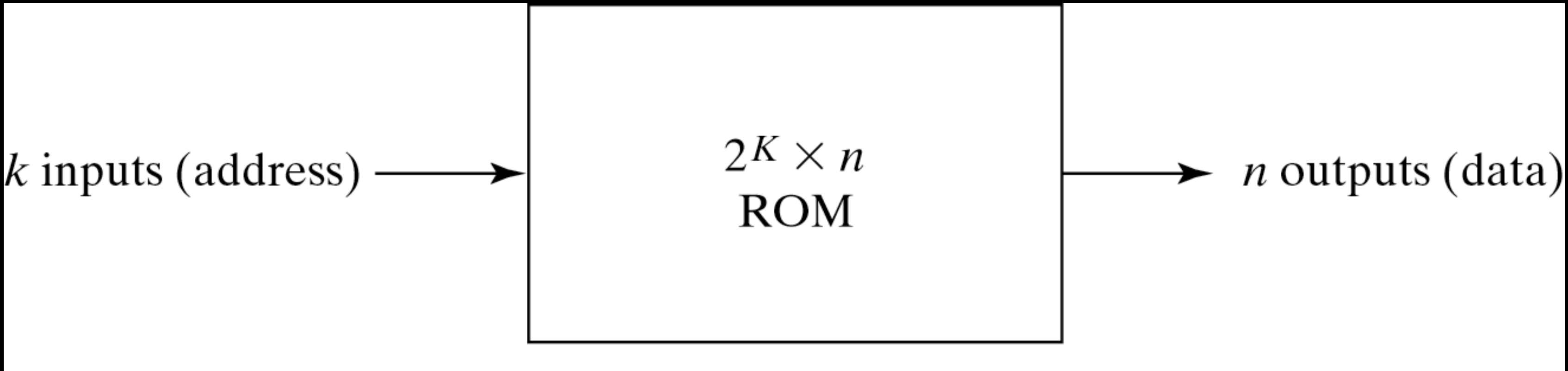
Read-Only Memory

8 bit data is called Byte, 16 bit is called Word

A block diagram of a ROM is shown below. It consists of k address inputs and n data outputs.

The number of words in a ROM is determined from the fact that k address input lines are needed to specify 2^k words.

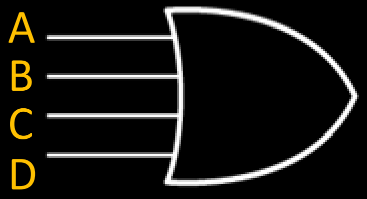
n represents the output data length



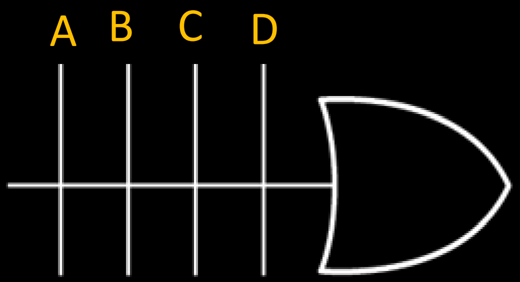
ROMs

- May have 100s of million gates interconnected through 100s of thousands of internal path

-To show internal logic of such a device – employ a special gate symbology applicable to array logic



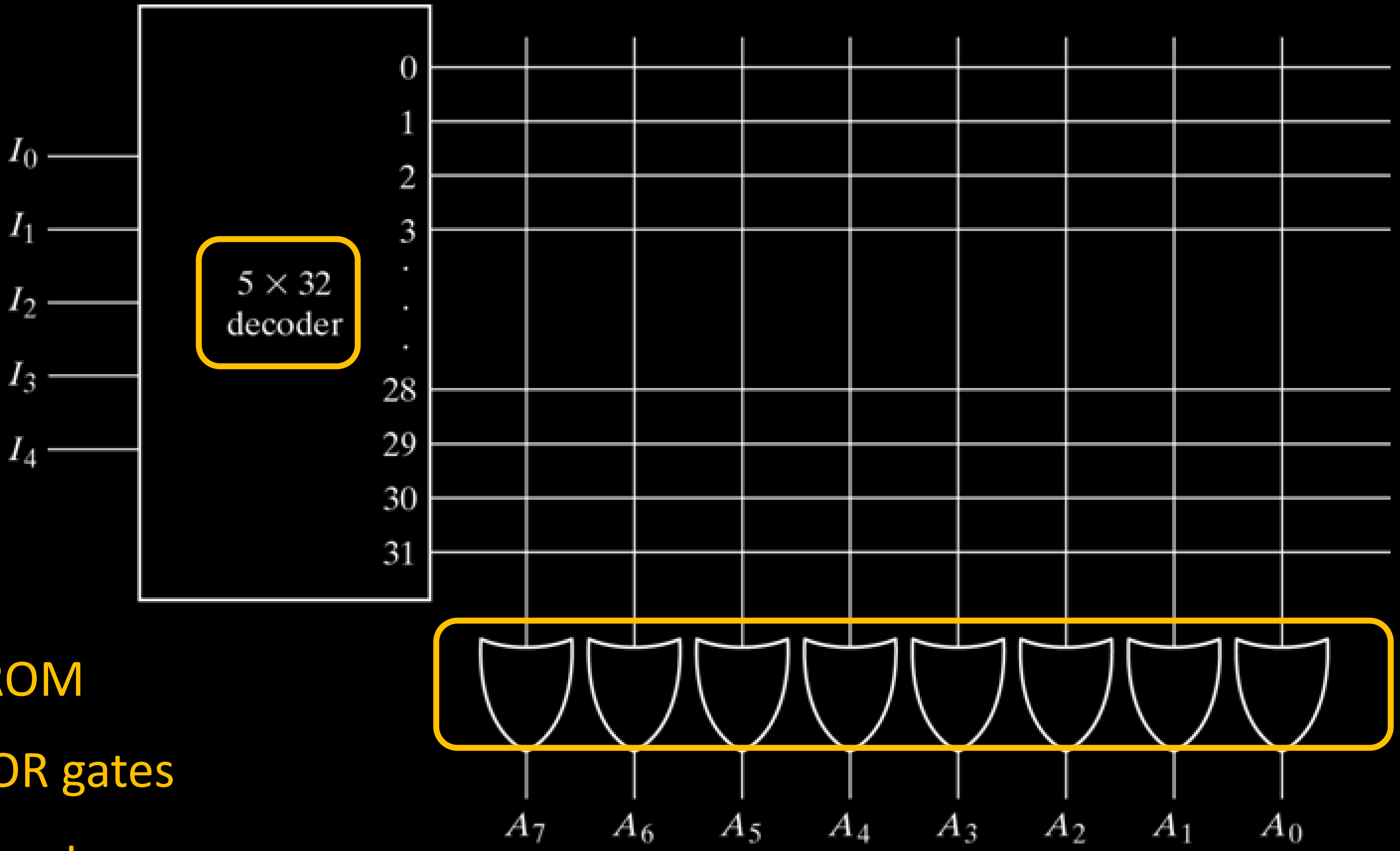
(a) Conventional symbol



(b) Array logic symbol

Read Only Memory

Internal Logic 32 X 8 ROM



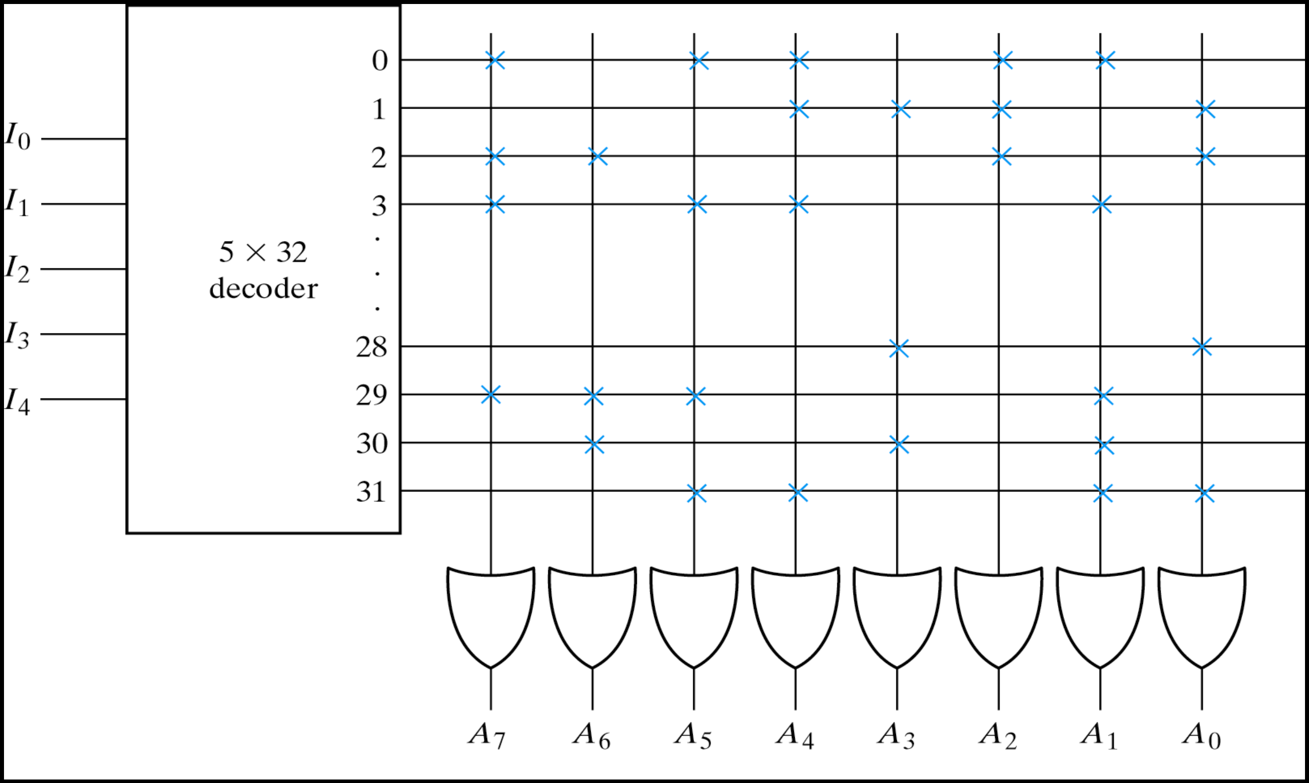
In General for 2^k X n ROM

K x 2^k Decoder and n OR gates

Each OR gate has 2^k inputs

Programming the ROM

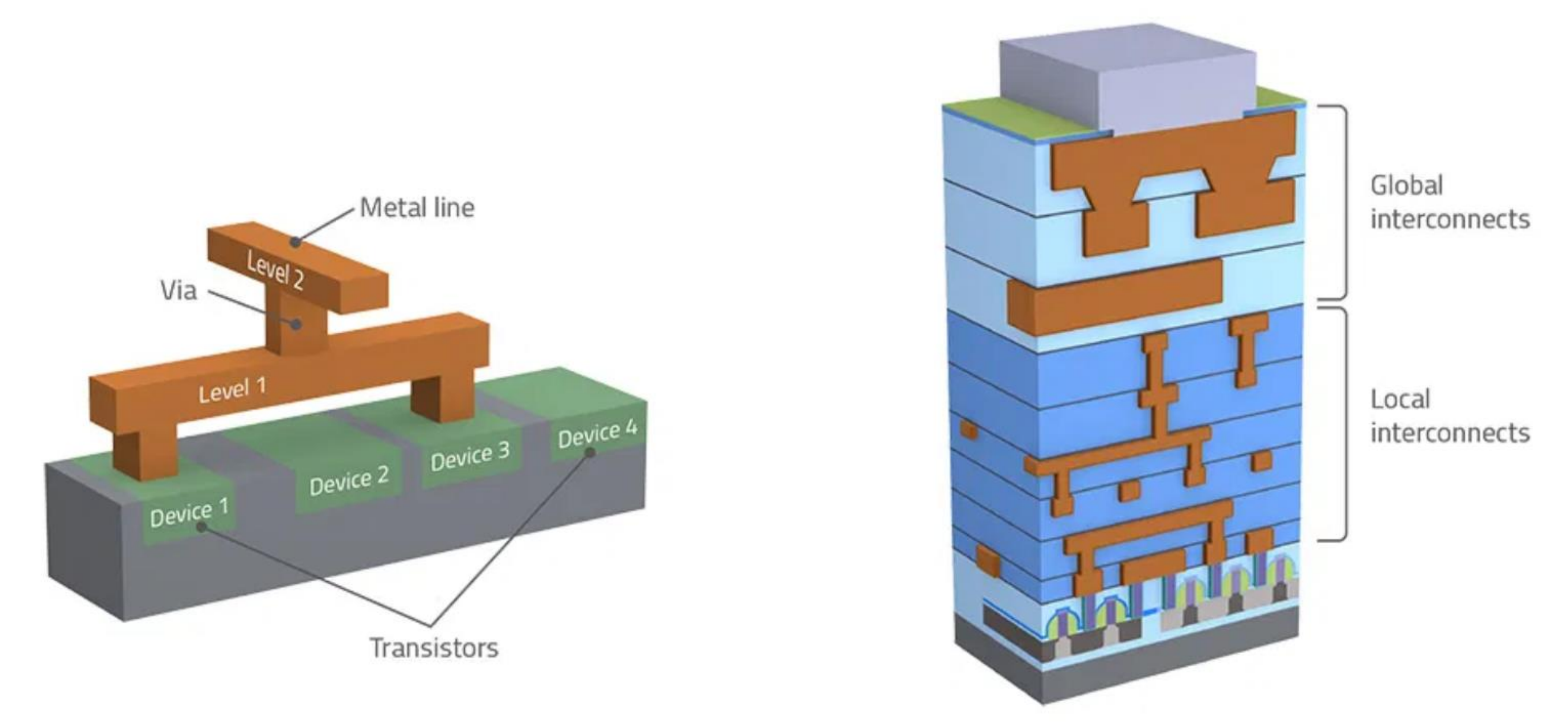
Address 3 = 10110010 is permanent storage using fuse link



X : means connection

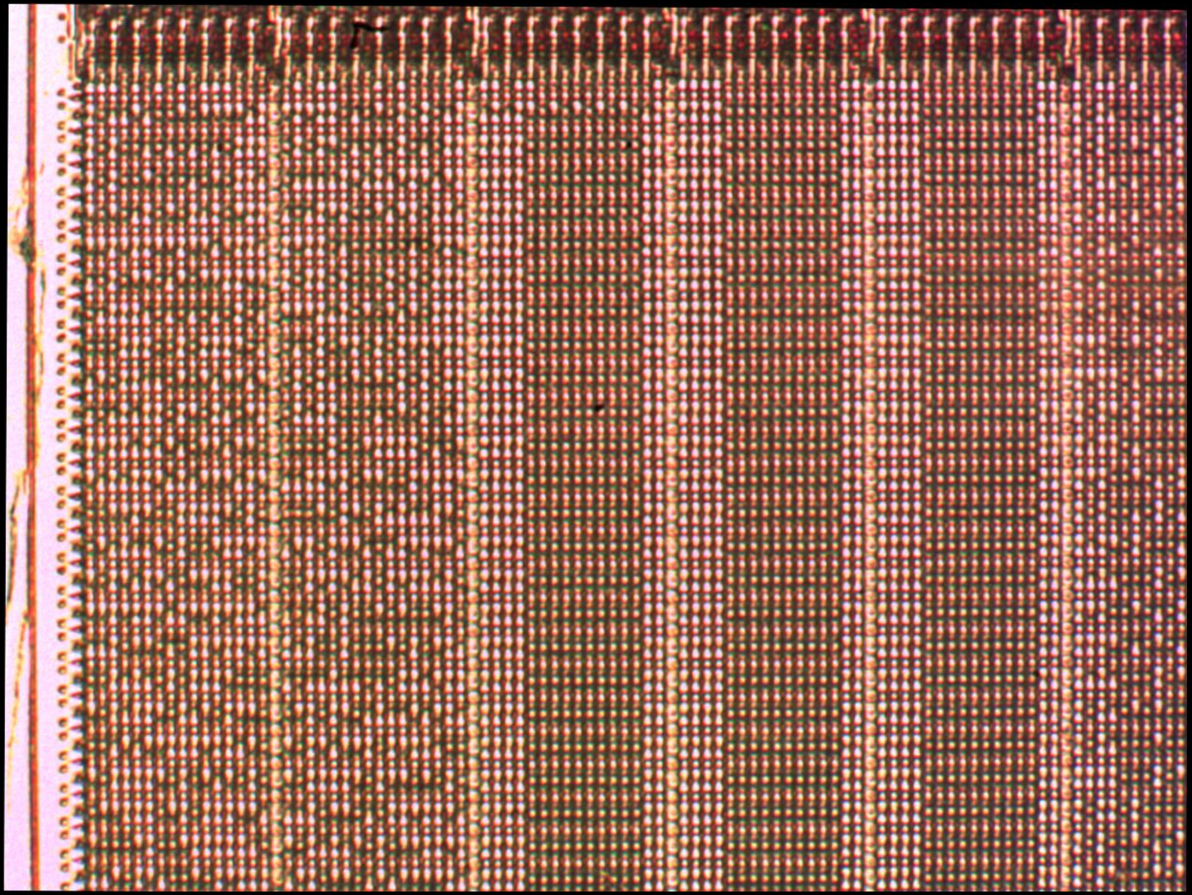
Programming the ROM

1. Masking During Metallization



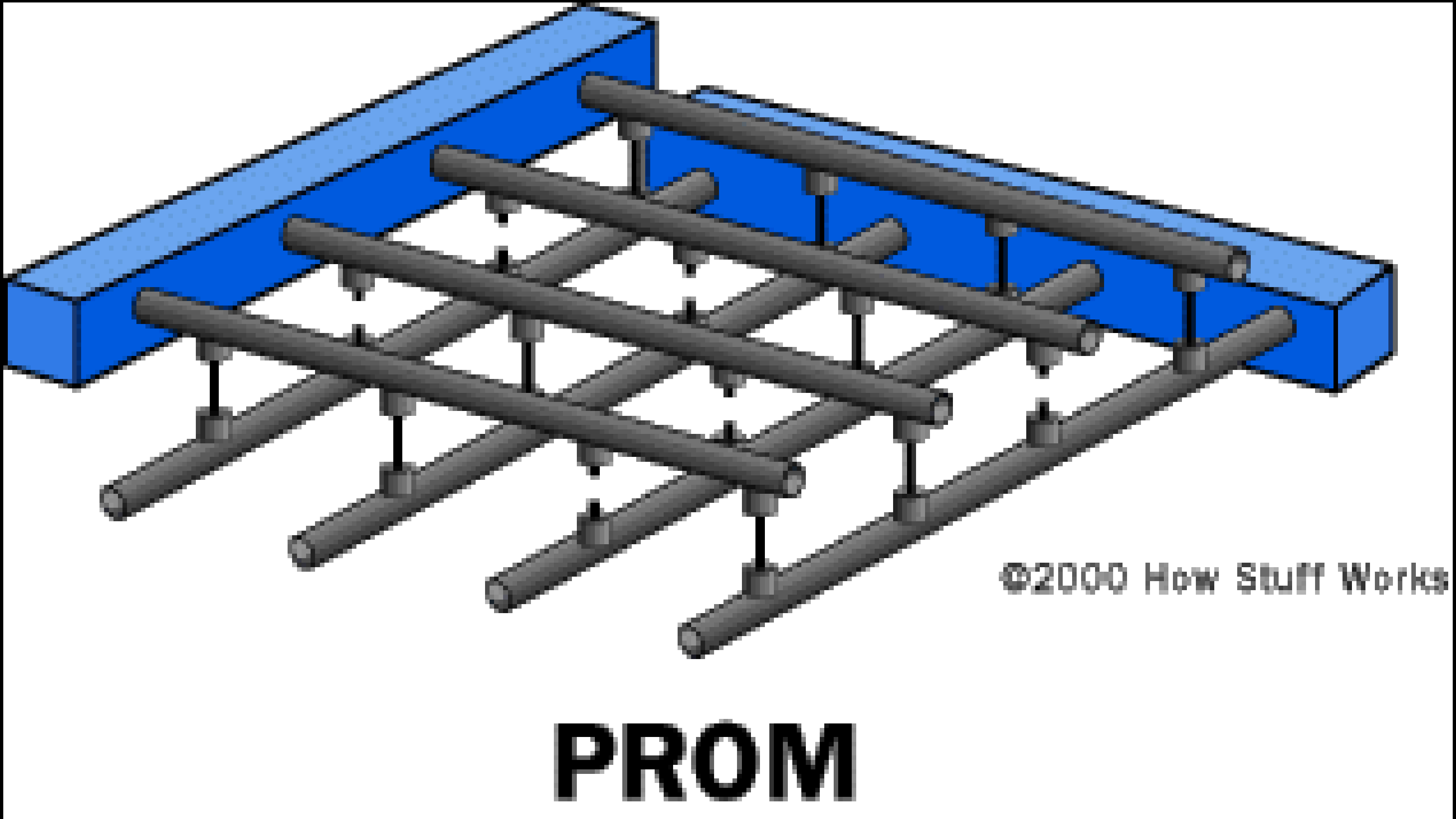
Programming the ROM

1. Masking During Metallization (PROM)



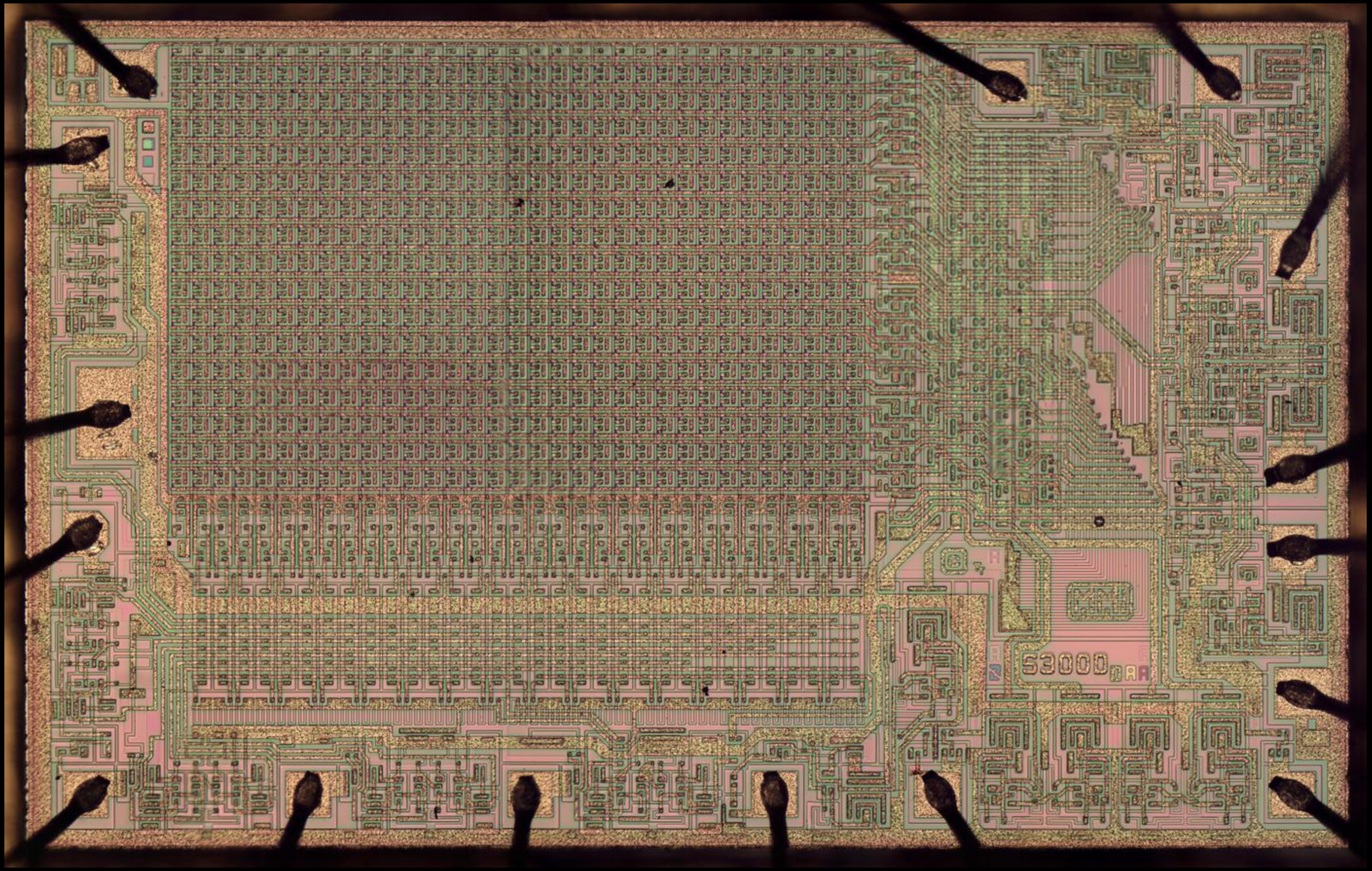
Programming the ROM

2. Fuse (PROM)



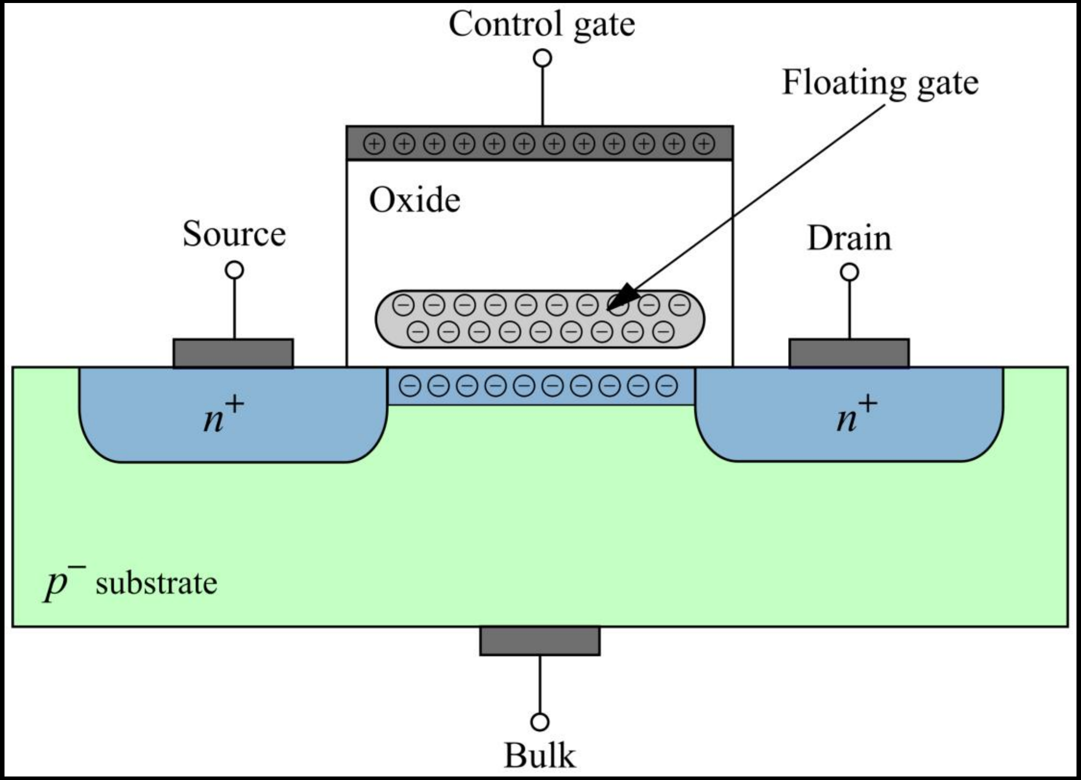
Programming the ROM

2. Fuse (PROM)



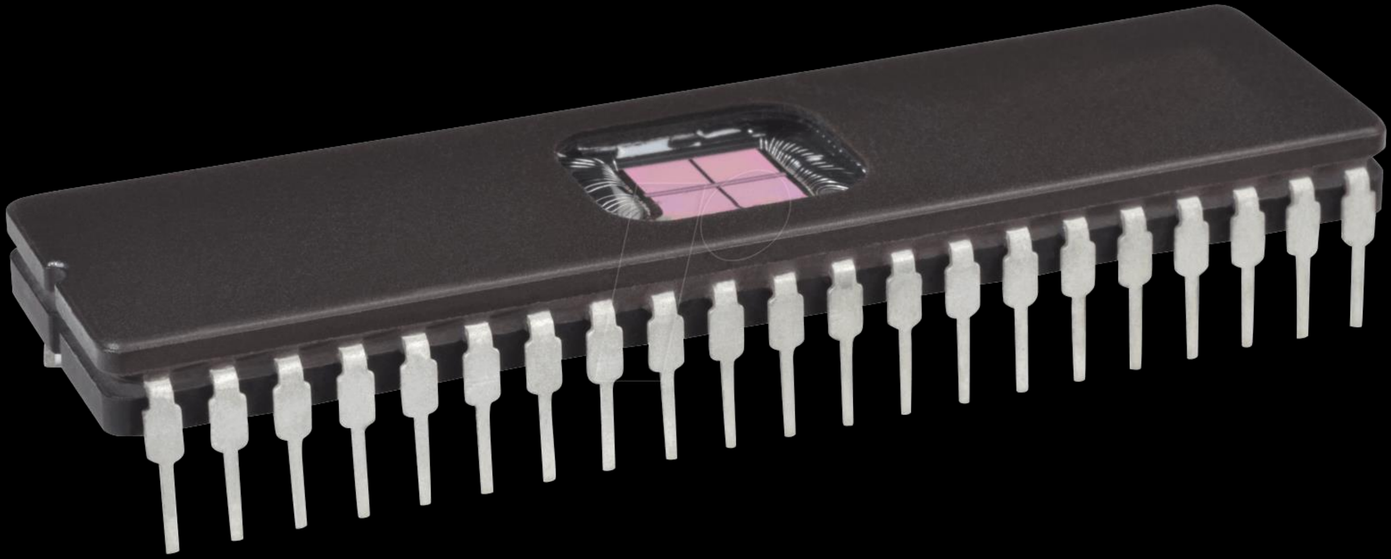
Programming the ROM

3. EPROM



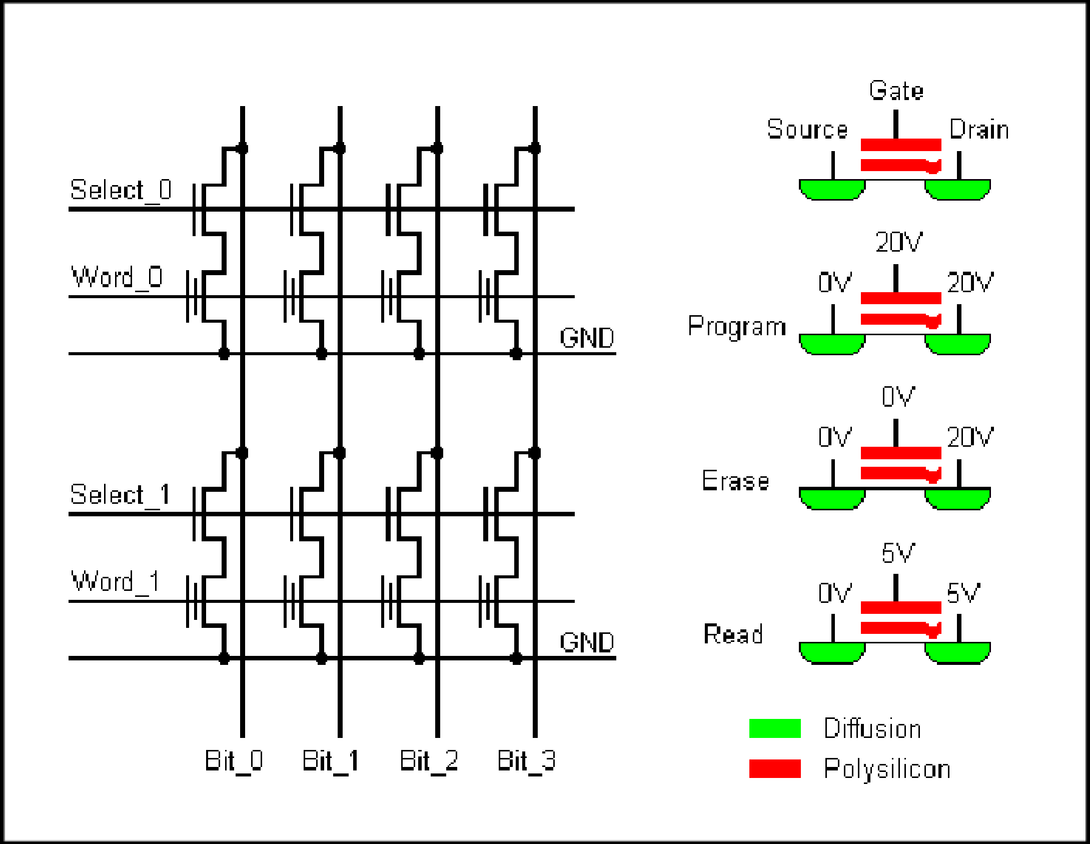
Programming the ROM

3. EPROM



Programming the ROM

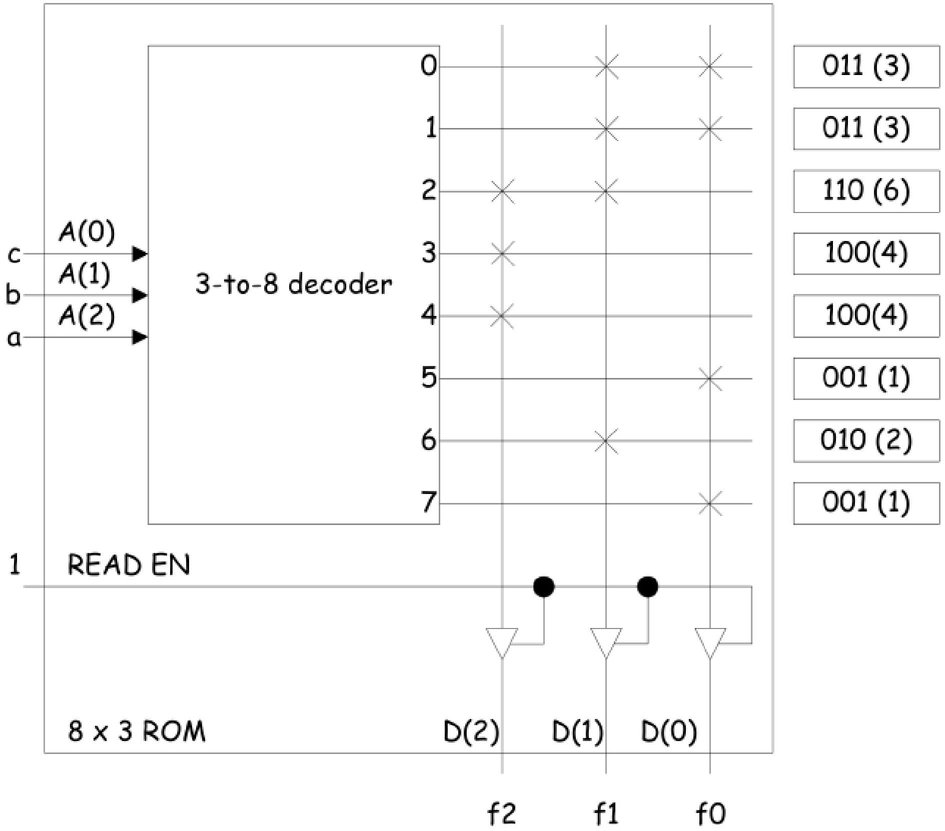
4. EEPROM



Programming the ROM

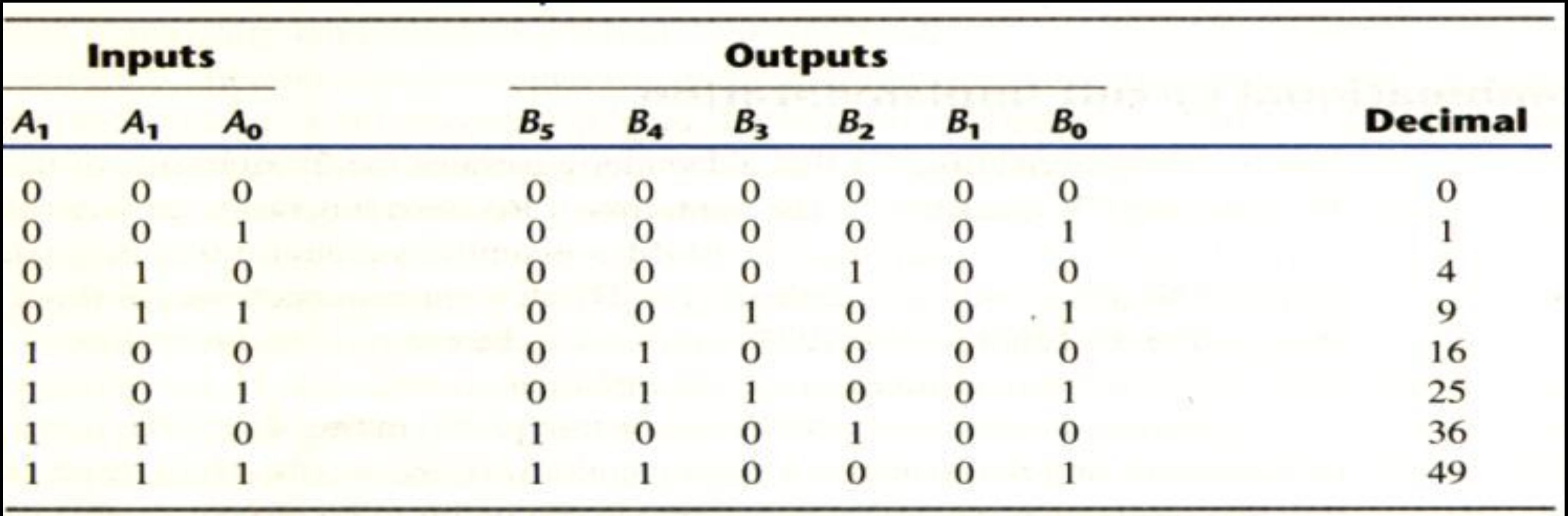
- □ E.g., Implement the 3-input logics $f_0 = \Sigma\,(0,1,5,7)$, $f_1 = \Sigma\,(0,1,2,6)$ and $f_2 = \Sigma\,(2,3,4)$ using a ROM.

a	b	c	f_2	f_1	f_0
0	0	0	0	1	1
0	0	1	0	1	1
0	1	0	1	1	0
0	1	1	1	0	0
1	0	0	1	0	0
1	0	1	0	0	1
1	1	0	0	1	0
1	1	1	0	0	1



Programming the ROM

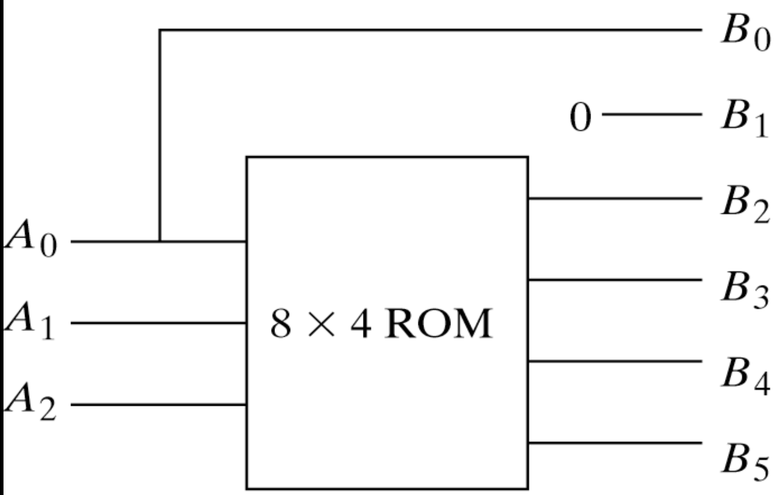
Example: Design a combinational circuit using a ROM. The circuit accepts a 3-bit number and generates an output binary number equal to the square of the input number.



Inputs			Outputs						
A_1	A_1	A_0	B_5	B_4	B_3	B_2	B_1	B_0	Decimal
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49

Programming the ROM

Example: Design a combinational circuit using a ROM. The circuit accepts a 3-bit number and generates an output binary number equal to the square of the input number.



(a) Block diagram

A_2	A_1	A_0	B_5	B_4	B_3	B_2
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	0	0

(b) ROM truth table

Thank You