Chapter 7 Memory and Programmable Logic

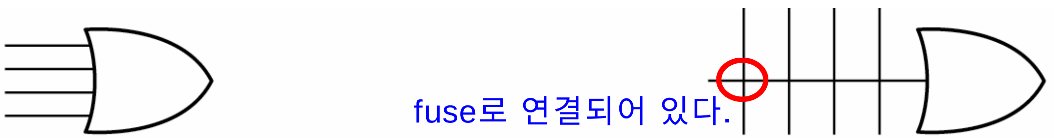
7-1 Introduction

- Memory unit
  - –a collection of cells capable of storing a large quantity of binary information and
    - to which binary information is transferred for storage
    - from which information is available when needed for processing
  - –together with associated circuits needed to transfer information in and out of the device
    - write operation: storing new information into memory
    - read operation: transferring the stored information out of the memory
- Two major types
  - –RAM (Random-access memory): Read + Write
    - accept new information for storage to be available later for use
  - –ROM (Read-only memory): perform only read operation (PLD)

FPGD : configured by downloading a stream of bits into the device to configure transmission gates to establish the internal connectivity required by a specified logic function

Programmable Logic Device

- Programmable logic device (PLD)
  - an integrated circuit with internal logic gates
    - hundreds to millions of gates interconnected through hundreds to thousands of internal paths
  - connected through electronic paths that behave similar to fuse
    - In the original state, all the fuses are intact
  - programming the device
    - blowing those fuse along the paths that must be removed in order to obtain particular configuration of the desired logic function
- Types
  - Read-only Memory (ROM, Section 7-5)
  - programmable logic array (PLA, Section 7-6) AND,OR에서 program 가능(납땜가능 x표시)
  - programmable array logic (PAL, Section 7-7) AND에서 program 가능(납땜가능 x표시)
  - field-programmable gate array (FPGA, Section 7-8)



(a) Conventional symbol

(b) Array logic symbol

Fig. 7-1 Conventional and Array Logic Diagrams for OR Gate

7-2 Random-Access Memory

- A memory unit stores binary information in groups of bits (word)
  - 1 byte = 8 bits
  - 16-bit word = 2 bytes, 32-bit word = 4 bytes

an entity of bits that move in and out of storage as a circuit

- Interface
  - n data input and output lines
  - k address selection lines
  - control lines specifying the direction of transfer

- Addressing
  - each word is assigned to an address
  - k-bit address: 0 to $2^k - 1$ word
  - size: K(kilo)=2^{10}, M(mega)=2^{20}, G(giga)=2^{30}
  - A decoder accepts an address and opens the paths needed to selection the word specified

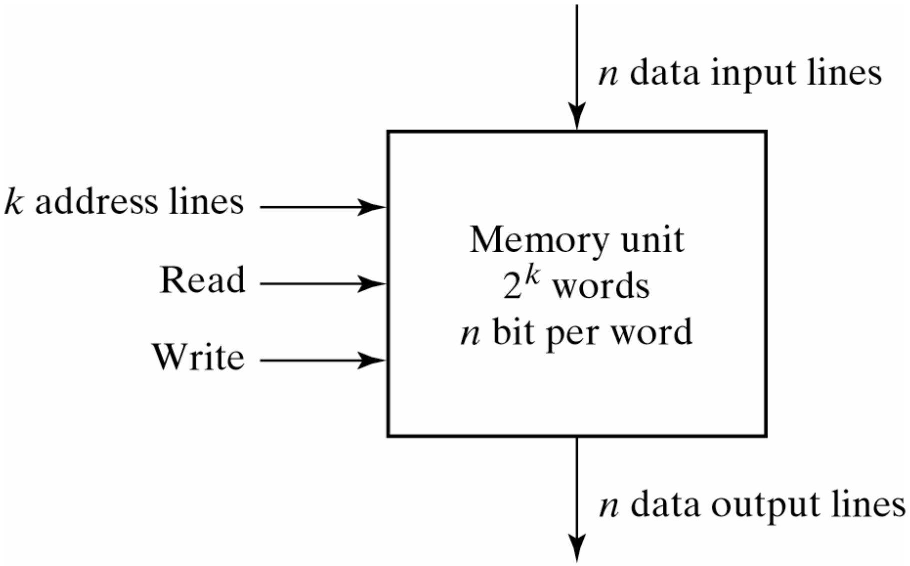


Fig. 7-2 Block Diagram of a Memory Unit

Example: 1K words of 16 bits

Capacity: 1K * 16 bits = 2K bytes = 2,048 bytes

Memory address		Memory contest
Binary	decimal	
0000000000	0	1011010101011101
0000000001	1	1010101110001001
0000000010	2	0000110101000110
	⋮	⋮
1111111101	1021	1001110100010100
1111111110	1022	0000110100011110
1111111111	1023	1101111000100101

Fig. 7-3 Content of a 1024 × 16 Memory

Addressing data: 16-bit data and 10-bit address

Write and Read Operations

- Steps of Write operation
  - –Apply the binary address to the address lines
  - –Apply the data bits to the data input lines
  - –Activate the write input
- Steps of Read operation
  - –Apply the binary address to the address lines
  - –Activate the read input
- Two ways of control inputs:
  - –separate read and write inputs
  - –memory enable (chip select) + Read/write (operation select)
    - widely used in commercial or multi-chip memory components

Table 7-1
Control Inputs to Memory Chip

Memory Enable	Read/Write	Memory Operation
0	X	None
1	0	Write to selected word
1	1	Read from selected word

Timing Waveforms of Memory

- Memory operation control: usually controlled by external devices such as CPU
  - CPU provides memory control signals to synchronize its internal clocked operations with memory operations
  - CPU also provides the address for the memory
- Memory operation times
  - ***access time***: time to select a word and read it
  - ***cycle time***: time to complete a write operation

  both must be within a time equal to a fixed number of CPU clock cycles

- The CPU is usually synchronized by its own clock. The memory, however, does not employ an internal clock. Instead, its read and write operations are specified by control inputs.

- The CPU must provide the memory control signals in such a way as to synchronize its internal clocked operations with the read and write operations of memory. This means that the access time and cycle time of the memory must be within a time equal to a fixed number of CPU clock cycles

Example

- 50 MHz CPU
  - 1 clock cycle = 20 ns
    - /50M sec
  - Read/write Op ≤ 50 ns
    - 50/20 = 2.5 or 3 cycles

- Memory enable and Read/Write signals must be activated after the signals in the address lines are stable to avoid destroying data in other memory words
- The two control signals must stay active for at least 50 ns
- The address and data signals must remain stable for a short time after the control signals are deactivated

- At the completion of the third clock cycle, the CPU can access the memory again with the next T1 cycle

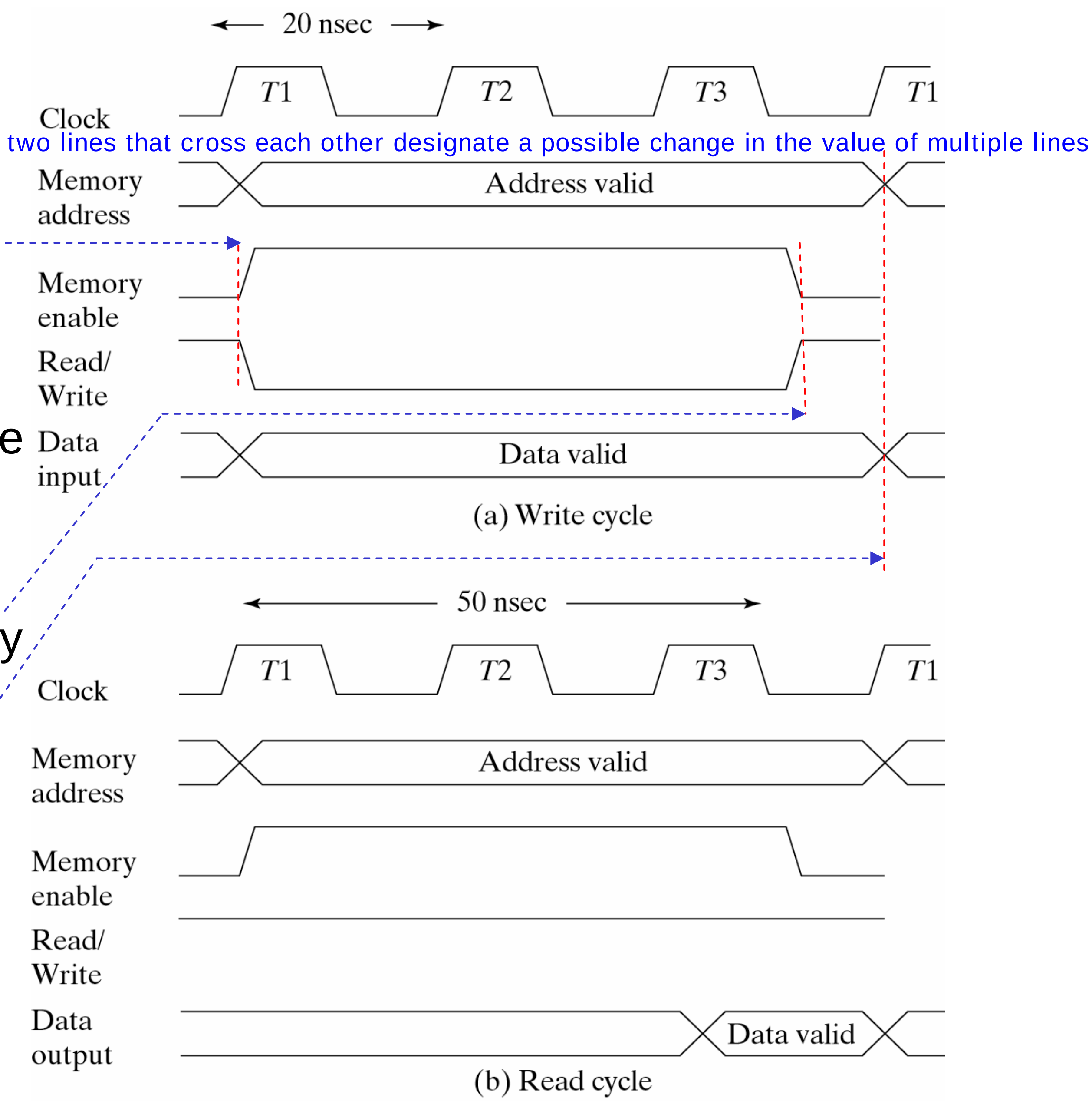


Fig. 7-4 Memory Cycle Timing Waveforms

Types of Memories

- Random vs. sequential
  - Random-Access Memory: each word is accessible separately
    - equal access time
  - Sequential-Access Memory: information stored is not immediately accessible but only at certain intervals of time
    - magnetic disk or tape
    - access time is variable depends on the position of the word with respect to the position of the read head
- Static vs. dynamic
  - SRAM: consists essentially of internal ***latches*** and remains valid as long as power is applied to the unit
    - advantage: shorter read and write cycles
  - DRAM: in the form of electric charges on ***capacitors*** which are provided inside the chip by MOS transistors
    - drawback: tend to discharge with time and must be periodically recharged by refreshing, cycling through the words every few ms
    - advantage: reduced power consumption and larger storage capacity
- Volatile vs. non-volatile
  - volatile: stored information is lost when power is turned off
  - Non-volatile: remains even after power is turned off
    - magnetic disk, flash memory
      - the data stored on magnetic components are represented by the direction of magnetization

7-3 Memory Decoding

- RAM of m words and n bits: m*n binary storage cells
- SRAM cell: stores one bit in its internal latch
  - SR latch with associated gates, 4-6 transistors

select	Read/write	operation
0	X	none
1	0	write
1	1	read

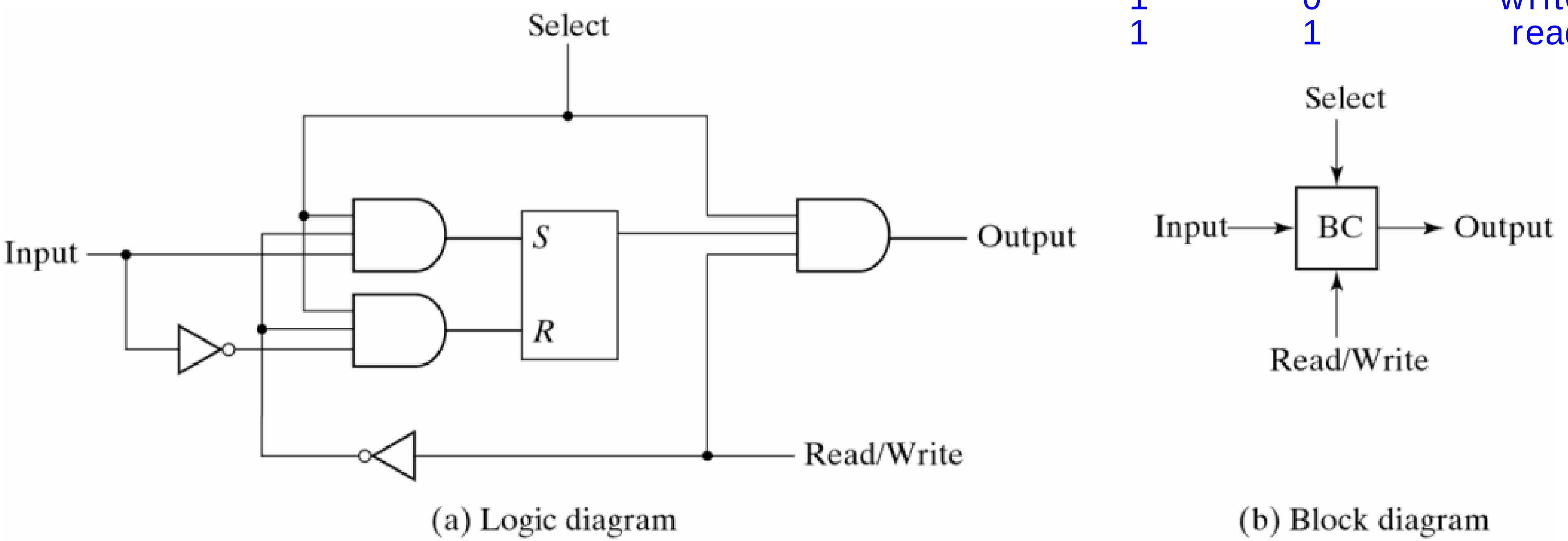


Fig. 7-5 Memory Cell

Example: capacity of 16 bits in 4 words of 4 bits each

- 2x4 decoder: select one of the 4 words
- enabled with the Memory enable signal
- Memory with 2^k words of n bits: k address lines go into a $k \times 2^k$ decoder

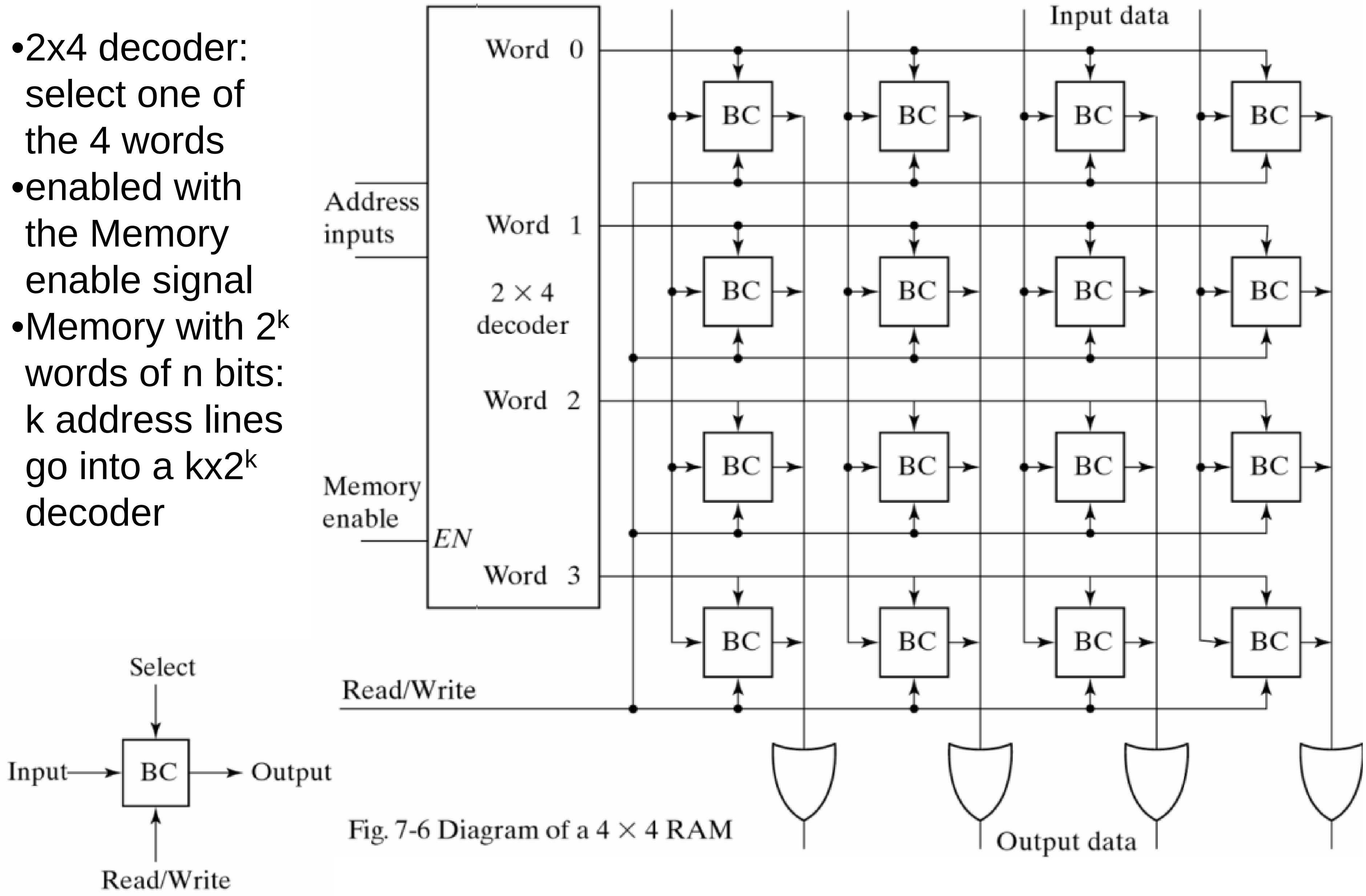


Fig. 7-6 Diagram of a 4 × 4 RAM

Coincident Decoding

- Decoder complexity: a decoder with k inputs and 2^k outputs requires 2^k AND gates with k inputs per gate
- 2-dimensional decoding: arrange cells in a square array

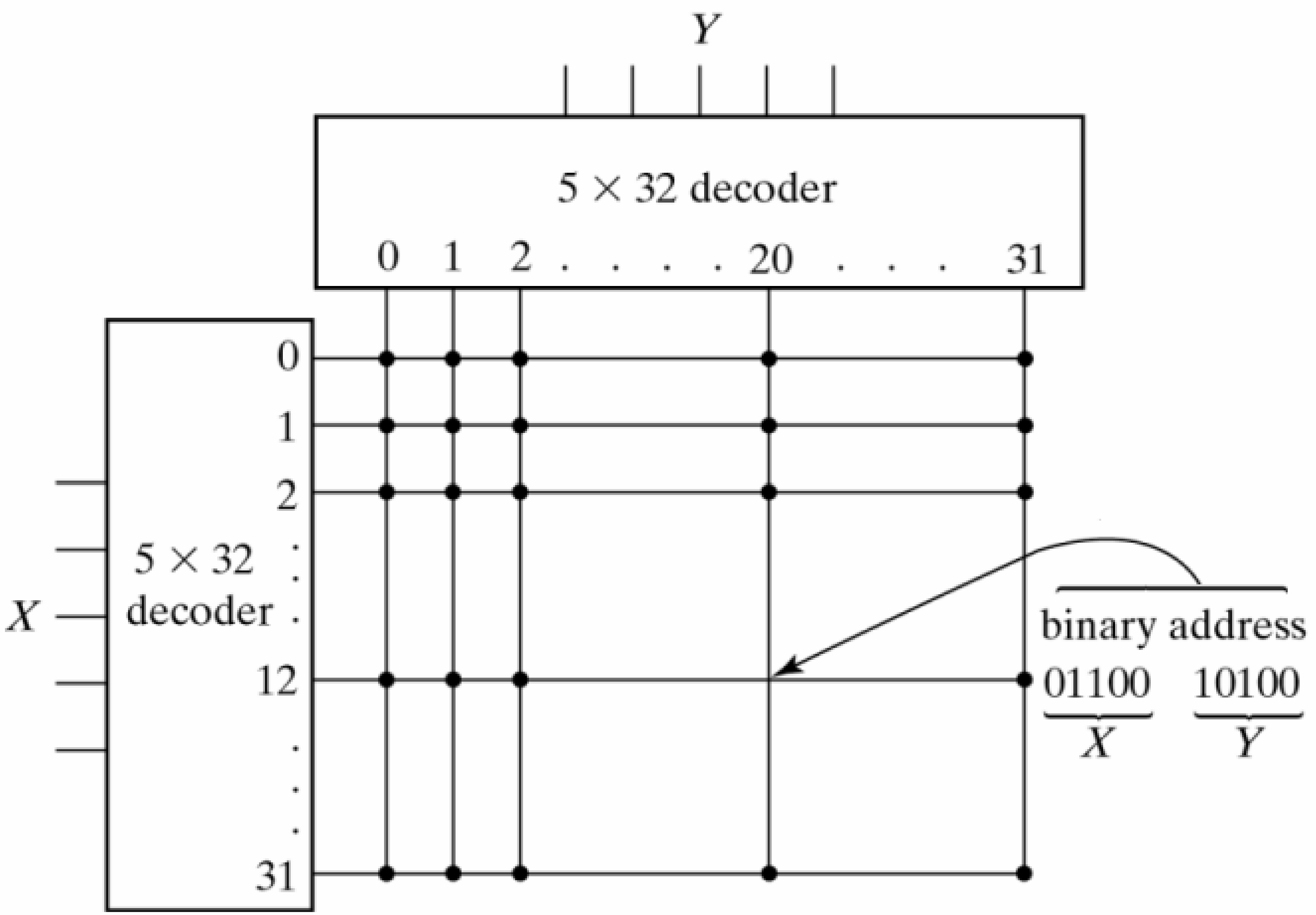


Fig. 7-7 Two-Dimensional Decoding Structure for a 1K-Word Memory

- 2 k/2-input decoders instead of 1 k-input decoder
  - one for row selection and the other for column selection
- 1K-word memory
  - a single 10x1,024 decoder: 1,024 10-input AND gates
  - two 5x32 decoders: 64 5-input AND gates

01100 10100 (404) ⇒ X=01100 (12) and Y=10100 (20)

Address Multiplexing

- DRAM: large capacity requires large address decoding
  - Simpler cell structure
    - DRAM: a MOS transistor and a capacitor per cell
    - SRAM: 6 transistors
  - Higher density: 4 times the density of SRAM
    - larger capacity
  - Lower cost per bit: 3-4 times less than SRAM
    - Lower power requirement
  - Preferred technology for large memories
    - 64K(=2^{16}) bits and 256M(=2^{28}) bits may need 16 and 28 address inputs
- ***Address multiplexing***: use a small set of address input pins to accommodate the address components
  - A full address is applied in multiple parts at different times
    - i.e. two-dimensional array: row address first and column address second
    - same set of pins is used for both parts
  - Advantage: reducing the number of pins for larger memory

Example: 64K-word memory

- 256 rows x 256 columns for $2^8 x 2^8 = 2^{16} = 64K$ words
- address strobes: enabling row and column address into their respective registers (no Memory enable)

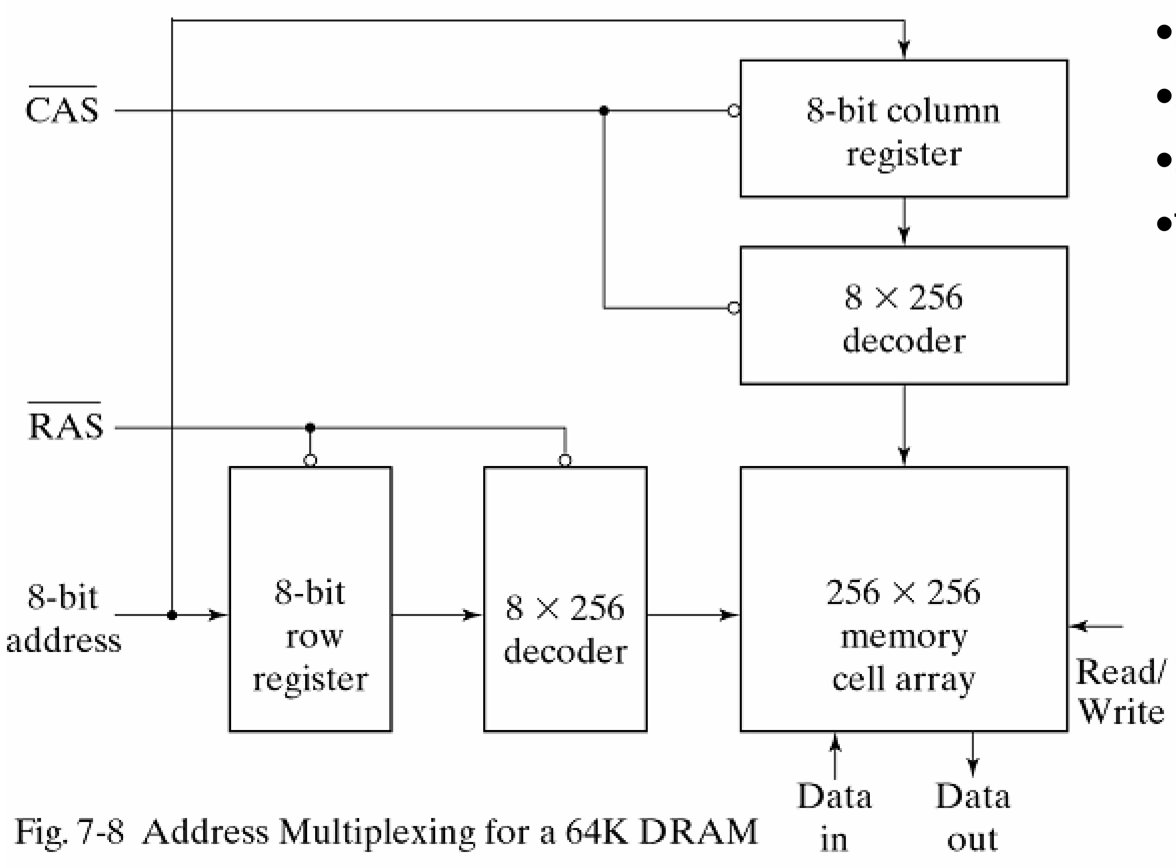


Fig. 7-8 Address Multiplexing for a 64K DRAM

- a single data input line
- a single data output line
- a Read/Write control
- two address strobes
  - RAS: enable 8-bit row register by level 0
  - CAS: enable 8-bit column register by level 0

7-5 Read-Only Memory

- ROM: permanent binary information is stored
  - –pattern is specified by the designer
  - –stays even when power is turned off and on again
- Pins
  - –k address inputs and n data outputs
  - –no data inputs since it doses not have a write operation
  - –one or more enable inputs

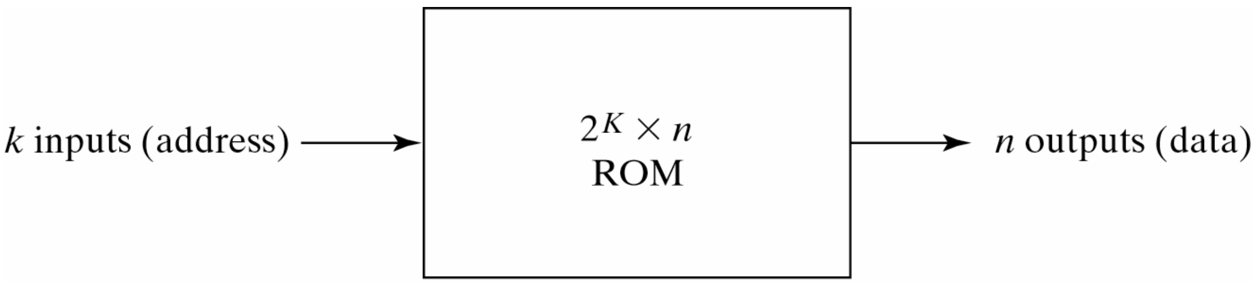


Fig. 7-9 ROM Block Diagram

Example: 32x8 ROM

- A 2^kxn ROM has an internal k x2^k decoder and n OR gates
- 32 words of 8 bits each
  - 32*8=256 programmable internal connections
  - 5 inputs decoded into 32 distinct outputs by 5x32 decoder
  - Each of 8 OR gates have 32 inputs

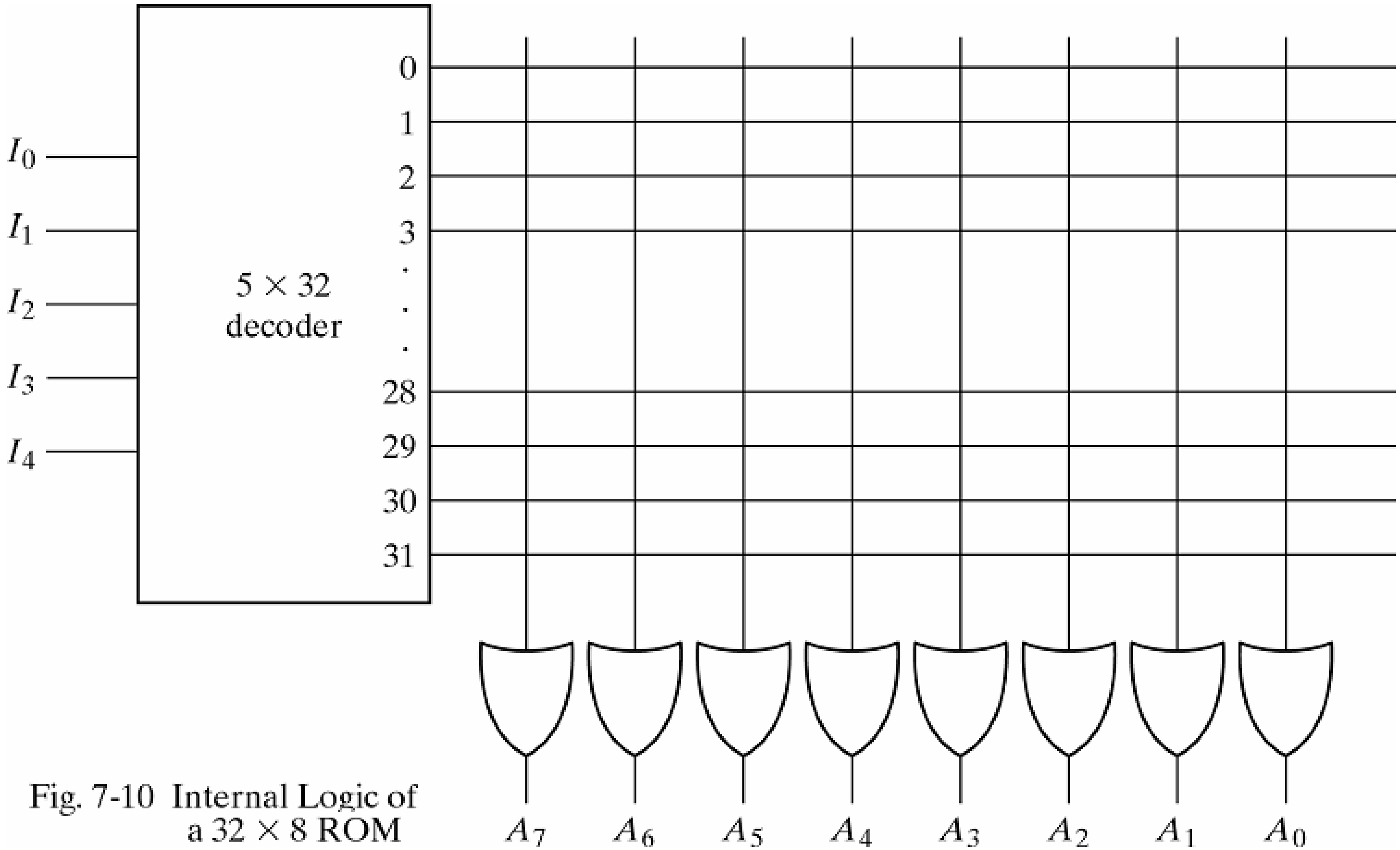


Fig. 7-10 Internal Logic of a 32 × 8 ROM

32x8 ROM

programmable intersection: crosspoint switch

- Two conditions
  - close: two lines are connected
  - open: two lines are disconnected
- Implemented by fuse
  - normally connects the two points
  - opened or "blown" by applying a high-voltage pulse

$A_7(I4,I3,I2,I1,I0)$
$=\Sigma(0,2,3,\ldots,29)$

Table 7-3 *ROM Truth Table (Partial)*

Inputs					Outputs							
I4	**I3**	**I2**	**I1**	**I0**	**A7**	**A6**	**A5**	**A4**	**A3**	**A2**	**A1**	**A0**
0	0	0	0	0	1	0	1	1	0	1	1	0
0	0	0	0	1	0	0	0	1	1	1	0	1
0	0	0	1	0	1	1	0	0	0	1	0	1
0	0	0	1	1	1	0	1	1	0	0	1	0
		⋮					⋮					
1	1	1	0	0	0	0	0	0	1	0	0	1
1	1	1	0	1	1	1	1	0	0	0	1	0
1	1	1	1	0	0	1	0	0	1	0	1	0
1	1	1	1	1	0	0	1	1	0	0	1	1

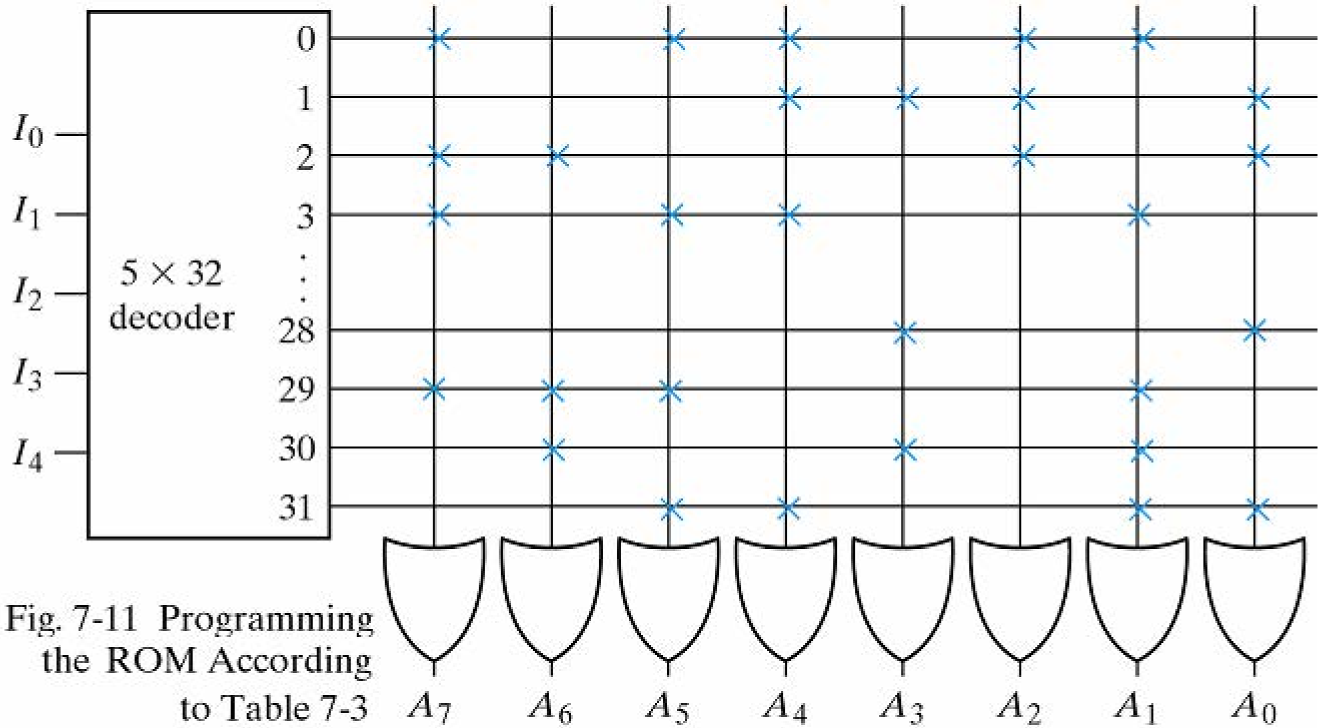


Fig. 7-11 Programming the ROM According to Table 7-3

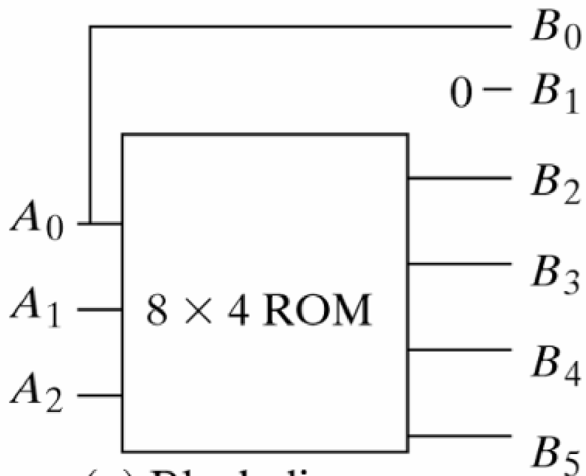
Combinational Circuit Implementation

- 2^kxn ROM: essentially a single device including both the decoder and the OR gates to generate any desired combinational circuit
  - k input address lines = k input variables
    - kx2^k decoder: generate 2^k minterms of the k inputs
  - n output data lines = n output functions
    - OR gates sum the minterms of Boolean functions
- Implementing combinational circuits by ROM needs only the ROM truth table
  - connect the crosspoints representing the minterms
  - no internal logic diagram is needed
- Procedures
  1. Determine the size of ROM
  2. Obtain the programming truth table
  3. 'Blow' the fuse pattern

Example 7-1 $f(x)=x^2$

Accept a 3-bit number and generate an output number equal to the square of the input number (Figure 7-12)

- 3 inputs and 6 outputs
- We can find that
  - Output B_0 is always equal to input A_0
  - output B_1 is always 0
- Minimum size ROM: 3 inputs and 4 outputs
  - 8x4 ROM



(a) Block diagram

Table 7-4 *Truth Table for Circuit of Example 7-1*

Inputs			Outputs						
A_2	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

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

Types of ROM

4 methods to program ROM paths

- mask programming ROM
  - customized and filled out the truth table by customer and masked by manufacturers during last fabrication process
  - costly; economical only if large quantities
- PROM: Programmable ROM
  - PROM units contain all the fuses intact initially
  - Fuses are blown by application of a high-voltage pulse to the device through a special pin by special instruments called PROM programmers
  - Written/programmed once; irreversible
- EPROM: erasable PROM
  - floating gates served as programmed connections
  - When placed under ultraviolet light, short wave radiation discharges the gates and makes the EPROM returns to its initial state
  - reprogrammable after erasure
- EEPROM: electrically-erasable PROM
  - erasable with an electrical signal instead of ultraviolet light
  - longer time is needed to write
  - flash ROM: limited times of write operations

Combinational PLDs

Combinational programmable logic device (PLD)

- programmable gates divided into an AND array and an OR array
- provide an AND-OR sum of product implementation

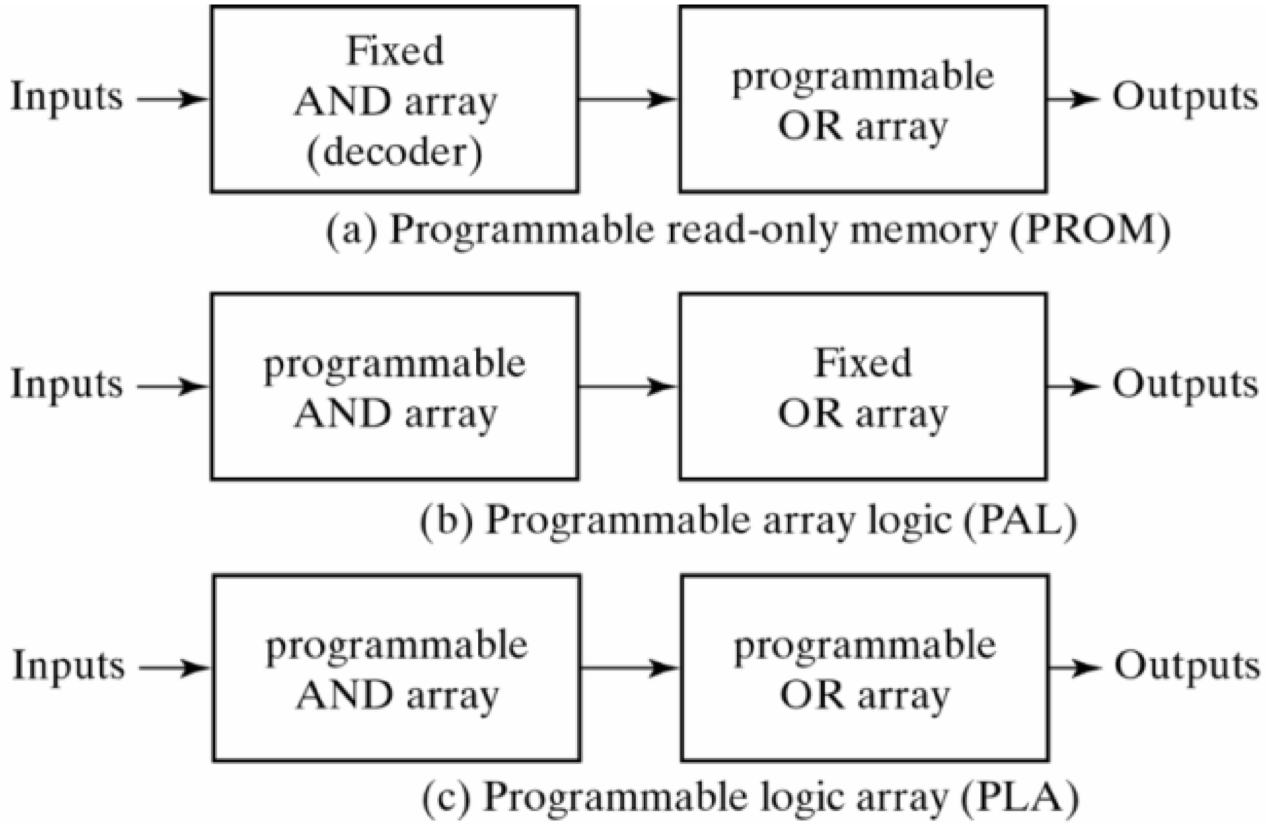


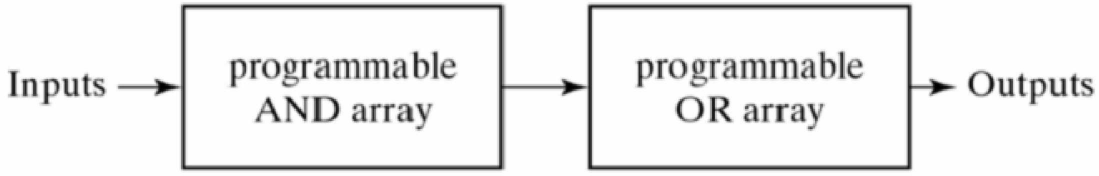
Fig. 7-13 Basic Configuration of Three PLDs

- a fixed AND array constructed as a decoder
- a programmable OR array to implement Boolean functions in sum of minterms

- a programmable AND array: to provide the product terms for Boolean functions

- both can be programmed
- most flexible

7-6 Programmable Logic Array



(c) Programmable logic array (PLA)

- Programmable Logic Array (PLA)
  - an array of programmable AND gates
    - can generate any product terms of the inputs
  - an array of programmable OR gates
    - can generate the sums of the products
  - only the needed product terms are generated (not all)
    - more flexible than ROM; use less circuits than ROM
- Size of PLA: specified by # of inputs, product terms and outputs
  - n inputs, k product terms and m outputs
  - n buffer-inverter gates, k AND gates, m OR gates, and m XOR gates
  - typical PLA may have 16 inputs, 48 product terms and 8 outputs
- Designing a digital system with a PLA
  - reduce the number of distinct product terms
  - the number of literals in a product is not important
- Implementing PLA
  - Mask programmable PLA: submit a PLA program table to the manufacturer
  - field programmable (FPLA): by commercial hardware programmer unit

PLA Example

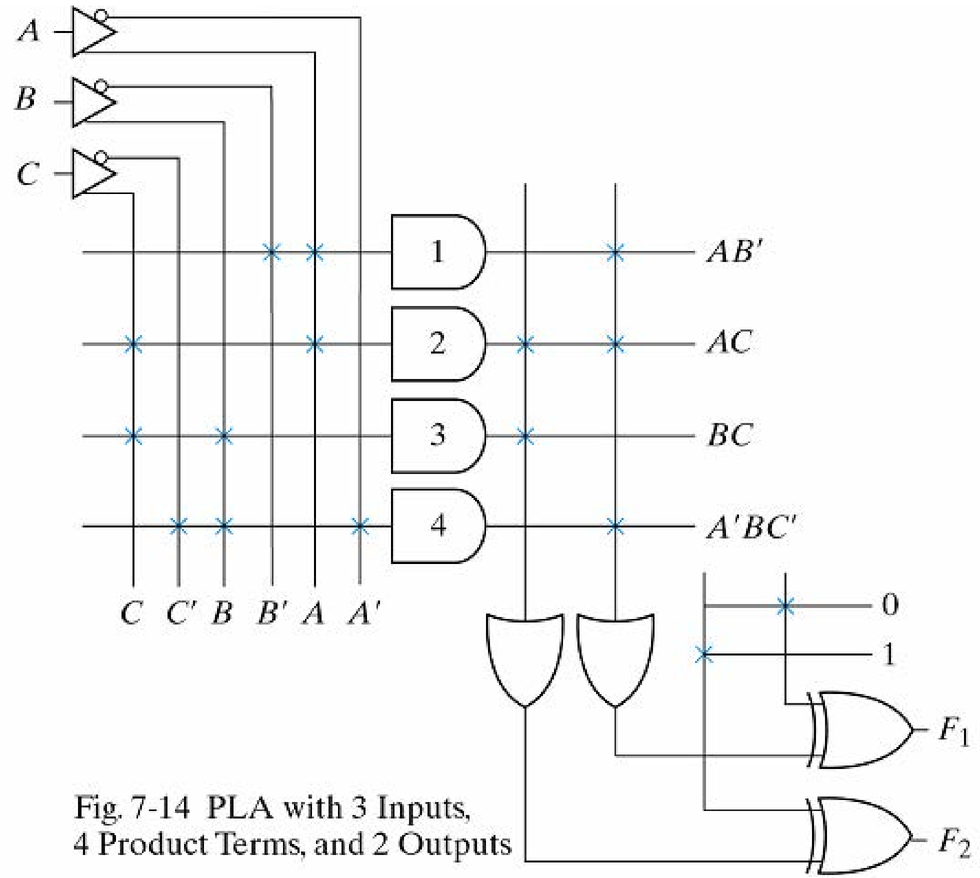


Fig. 7-14 PLA with 3 Inputs, 4 Product Terms, and 2 Outputs

- Example: AND/OR/XOR

  F1 = AB′ + AC + A′BC′

  F2 = (AC + BC)′

  XOR gates can invert the outputs
  - invert: connected to 1
  - not change: connected to 0

Table 7-5 *PLA Programming Table*

	Product Term	Inputs A	B	C	Outputs (T) F_1	(C) F_2
AB′	1	1	0	–	1	–
AC	2	1	–	1	1	1
BC	3	–	1	1	–	1
A′BC′	4	0	1	0	1	–

PLA programming table: 3 sections

1. list the product terms
2. specify the required paths between inputs and AND gates
3. specify the paths between the AND and OR gates

Specifying the fuse map and submitted to the manufacturer

Example 7-2

Implement: $F_1(A, B, C) = \Sigma\,(0, 1, 2, 4)$; $F_2(A, B, C) = \Sigma\,(0, 5, 6, 7)$

1. Simply both the true and complement of the functions in sum of products

2. Find the combination with minimum number of product terms

   $F_1=(AB+AC+BC)'$

   $F_2=AB+AC+A'B'C'$

A \ BC	00	01	11	10
0	1	1	0	1
1	1	0	0	0

$F_1 = A'B' + A'C' + B'C'$

$F_1 = (AB + AC + BC)'$

A \ BC	00	01	11	10
0	1	0	0	0
1	0	1	1	1

$F_2 = AB + AC + A'B'C'$

$F_2 = (A'C + A'B + AB'C')'$

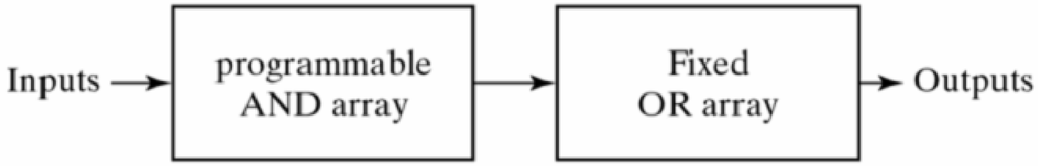
3. Obtain the PLA programming table

PLA programming table

	Product term	Inputs A	B	C	Outputs (C) F_1	(T) F_2
AB	1	1	1	–	1	1
AC	2	1	–	1	1	1
BC	3	–	1	1	1	–
$A'B'C'$	4	0	0	0	–	1

Fig. 7-15 Solution to Example 7-2

7-7 Programmable Array Logic



(b) Programmable array logic (PAL)

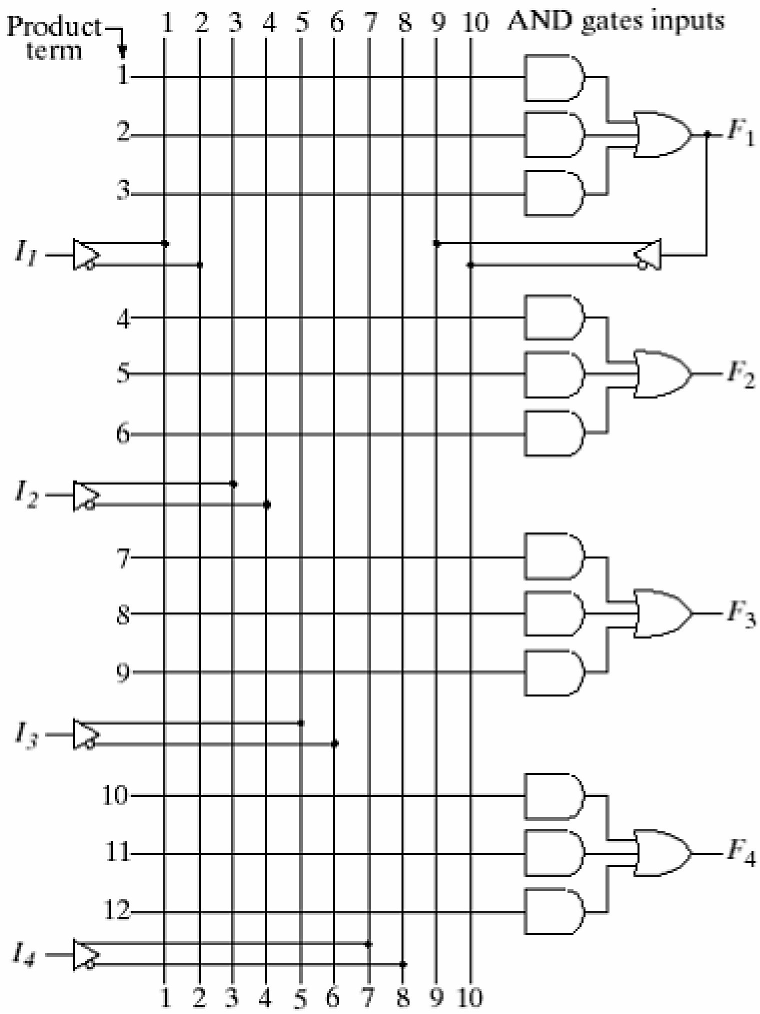
PAL: a programmable AND array and a fixed OR array

– easier to program, but not as flexible as PLA

Example: PAL with 4 inputs, 4 outputs, and 3-wide AND-OR structure (Figure 7-16)

- each input has a buffer-inverter gate
- each output is generated by a fixed OR gate
- 4 sections of 3-wide AND-OR array
  – each AND gate has 10 programmable input connections

- A typical PAL may have 8 inputs, 8 outputs, and 8 sections, each consisting of an 8-wide AND-OR array
  – May use two sections to implement a large Boolean function
- Product terms cannot be shared
  – Each function is simplified itself



Example: PAL Implementation

- Implement the following functions
  - $w(A,B,C,D) = \Sigma(2,12,13)$
  - $x(A,B,C,D) = \Sigma(7,8,9,10,11,12,13,14,15)$
  - $y(A,B,C,D) = \Sigma(0,2,3,4,5,6,7,8,10,11,15)$
  - $z(A,B,C,D) = \Sigma(1,2,8,12,13)$
- Simplify the functions
  - $w = ABC' + A'B'CD'$
  - $x = A + BCD$
  - $y = A'B + CD + B'D'$
  - $z = ABC' + A'B'CD' + AC'D' + A'B'C'D$
  - $\quad = w + AC'D' + A'B'C'D$

Table 7-6 PAL Programming Table

Product Term	AND Inputs A	B	C	D	W	Outputs
1	1	1	0	–	–	$w = ABC'$
2	0	0	1	0	–	$+ A'B'CD'$
3	–	–	–	–	–	
4	1	–	–	–	–	$x = A$
5	–	1	1	1	–	$+ BCD$
6	–	–	–	–	–	
7	0	1	–	–	–	$y = A'B$
8	–	–	1	1	–	$+ CD$
9	–	0	–	0	–	$+ B'D'$
10	–	–	–	–	1	$z = w$
11	1	–	0	0	–	$+ AC'D'$
12	0	0	0	1	–	$+ A'B'C'D$

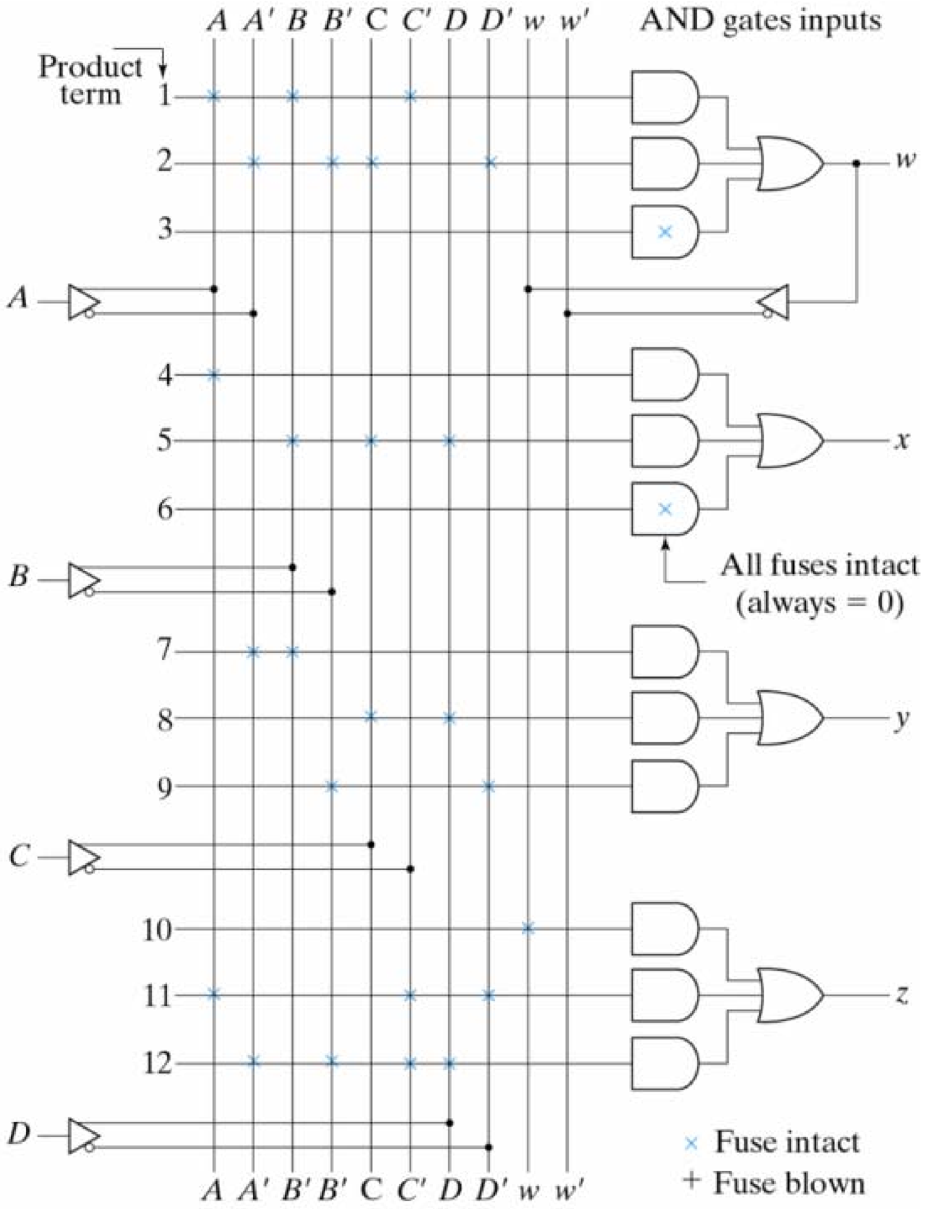


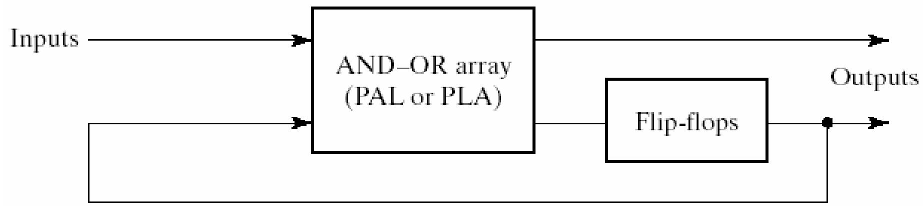
Fig. 7-17 Fuse Map for PAL as Specified in Table 7-6

7-8 Sequential Programmable Devices

- Sequential programmable devices
  - combinational PLD + flip-flops
  - perform a variety of sequential-circuit functions
- Three major types
  - Sequential (or simple) programmable logic device (SPLD)
    - field-programmable logic sequencer (FPLS)
  - Complex programmable logic device (CPLD)
  - Field programmable gate array (FPGA)

Many commercial vendor-specific variants and internal logic of these devices is too complex to be shown here

SPLD – Simple / Sequential PLD



- SPLD includes flip-flops and AND-OR array
  - flip-flops connected to form a register
  - FF outputs could be included in product terms of AND array
  - Field-programmable logic sequencer (FPLS)
    - first programmable device developed, FF may be of D or JK type
    - not succeed commercially due to too many programmable connections
  - Combinational PAL together with D flip-flops: most used
- Macrocell: a section of an SPLD
  - a circuit containing a sum-of-products combinational logic function and an optional flip-flop
  - a typical SPLD contains 8-10 macrocells
  - Features:
    - programming AND array
    - use or bypass the flip-flop
    - select clock edge polarity
    - preset or clear for the register
    - complement an output

✓ FF is connected to a common clock

✓ OE (output enable) signal also controls all the three-state buffers

✓ FF output is fed back to PAL inputs

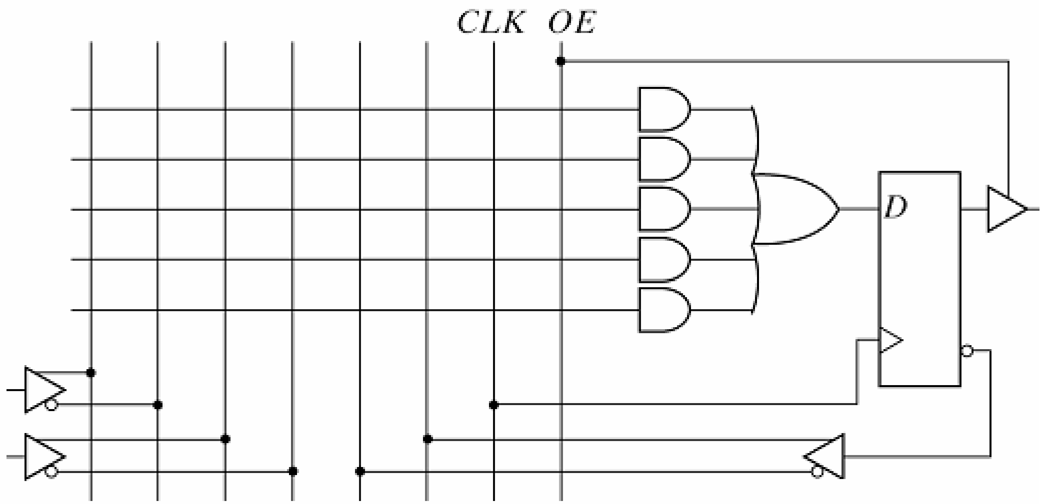


Fig. 7-19 Basic Macrocell Logic

CPLD - Complex Programmable Logic Device

- CPLD: a collection of PLDs to be connected to each other through a programmable switch matrix
  - input/output blocks provide connections to IC pins
  - each I/O pin is driven by a three-state buffer and can be programmed to act as input or output
  - switch matrix receives inputs from I/O block and directs it to individual macrocells
  - selected outputs from macrocells are sent to the outputs as needed
  - each PLD typically contains from 8 to 16 macrocells

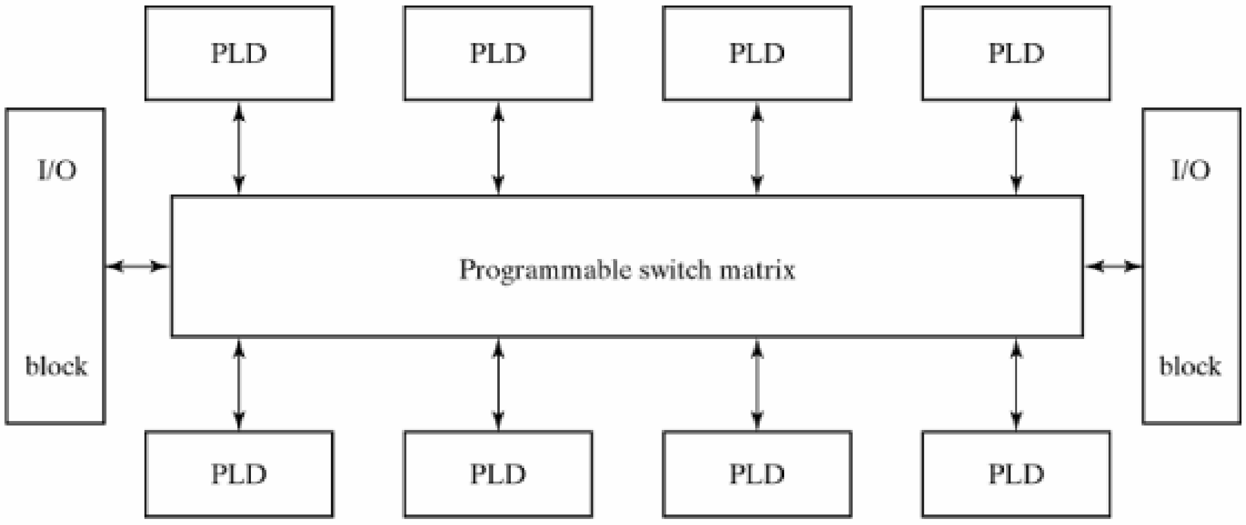
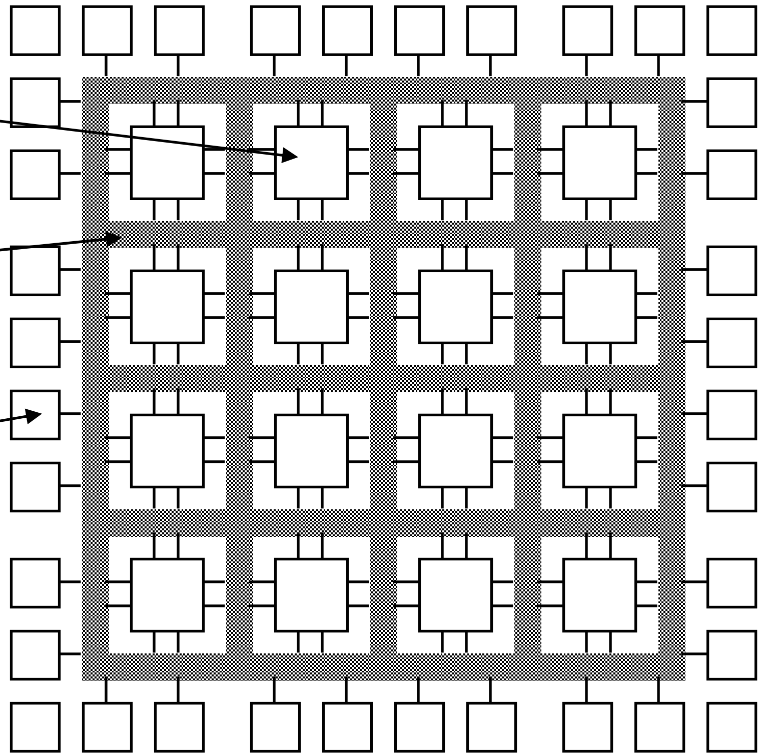


Fig. 7-20 General CPLD Configuration

FPGA – Field-Programmable Gate Array

- gate array: basic component used in VLSI
  - consist of a pattern of gates fabricated in an area of silicon and repeated thousands of times
- FPGA: an array of hundreds or thousands of logic blocks
  - surrounded by programmable input and output blocks
  - connected together via programmable interconnections
  - a logic block consists of look-up tables, multiplexers, gates, and flip-flops
    - look-up table: a truth table stored in a SRAM and providing combinational circuit functions for the logic block
    - SRAM instead of ROM
      - advantage: the table can be programmed
      - drawback: memory is volatile, reload/reprogram required after power on again
- Complexity
  - PALs, PLAs = 10 - 100 Gate Equivalents
  - FPGAs = 100 - 1000(s) of Gate Equivalents

Field-Programmable Gate Arrays



- Logic blocks
  - To implement combinational and sequential logic
- Interconnect
  - Wires to connect inputs and outputs to logic blocks
- I/O blocks
  - Special logic blocks at periphery of device for external connections
- Key questions:
  - How to make logic blocks programmable?
  - How to connect the wires?
  - *After the chip has been fabbed*

Summary

Chapter 7 Memory and Programmable Logic

7-1 Introduction

7-2 Random-Access Memory

7-3 Memory Decoding

7-4 Error Detection and Correction

7-5 Read-Only Memory

7-6 Programmable Logic Array

7-7 Programmable Array Logic

7-8 Sequential Programmable Devices