COEN 212: DIGITAL SYSTEMS DESIGN I Lecture 12: Memory and Programmable Logic Devices

Instructor: Dr. Reza Soleymani, Office: EV-5.125, Telephone: 848-2424 ext.: 4103.

Lecture 12: Objectives of this lecture

- In this lecture, we talk about:
  - Random Access Memory (RAM),
  - Read Only Memory (RMO),
  - Memory Decoding,
  - Programmable Logic Array (PLA),
  - Programmable Array Logic (PAL).

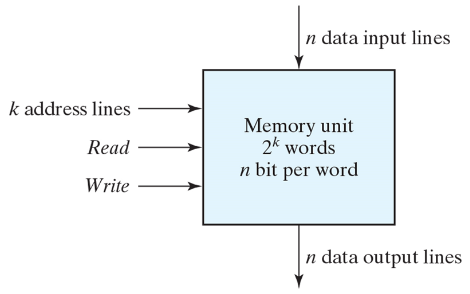
- Revisiting Error Detection and Correction.

Lecture 12:
Reading for this lecture

- Digital Design by M. Morris R. Mano and Michael D. Ciletti, 6th Edition, Pearson, 2018:
  - Chapter 7

Lecture 12: Random Access Memory (RAM):

- Block Diagram of RAM:



- If a RAM has k address lines it can point to 2^k words.
- Example:
  - $2^{10} = 1024 \approx 1000$. So, 10 address lines → 1 k words.
  - $2^{20} \approx 10^6$. So, 10 address lines → 1 M words.
  - If the width is 8, each word is called a byte.
- A read command (say read input high) puts the content of the location specified by the k address bits on the output lines.
- Write command allows the n bits placed on the input lines be written in the location addressed by the k addresses bits.

Lecture 12: Random Access Memory (RAM):

- Example: Entries of a 1024×16 RAM:

Address	Content
0000000000	1011011010011100
0000000001	1100100111100010
0000000010	0011110101111010
⋮	⋮
1111111101	1001110100001001
1111111110	0101111100011101
1111111111	1010110011101011

- To Write:
  - Apply the address of the location to the address lines.
  - Apply the data to be written to the input lines.
  - Activate the write input.
- To Read:
  - Apply the address to the address lines.
  - Activate the write input. Data will be available on the output lines.

Lecture 12: Random Access Memory (RAM):

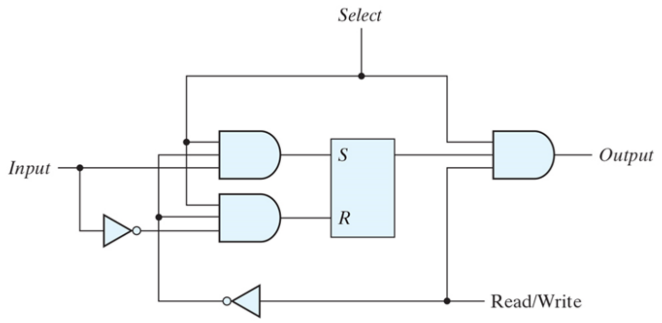
- RAM with Chip Enable:

Memory Enable	Read/Write	Memory Operation
0	X	None
1	0	Write
1	1	Read

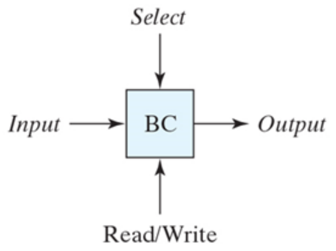
- There are two types of RAM:
  - Static RAM (SRAM): value of bits are the state Latches.
  - Dynamic Ram (DRAM): value of bits are the changes on capacitors.
- DRAM: More dense, less power.
- SRAM: faster (shorter read ad write time). uses a single transistor per bit instead of several transistors in SRAM, it is more dense (has more storage capacity) and also consumes less power.

Lecture 12: Internal Construction of RAM:

- A RAM consists a 2^k rows of n memory cells.
- Each memory cell has: An input, An output, A select (enable) input and a read/write input
- The logic diagram of a memory cell is:



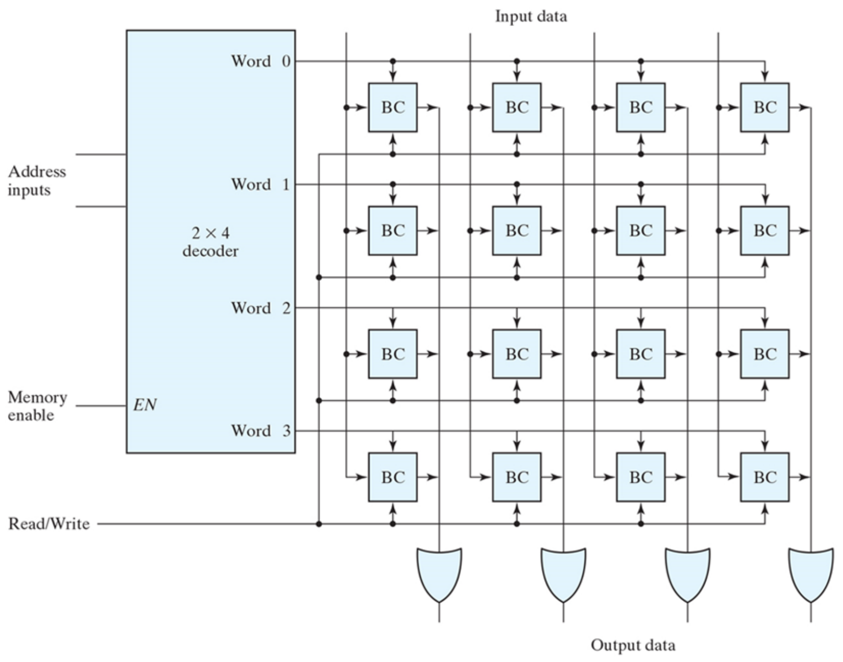
- The block diagram is:



Lecture 12: Address Decoding:

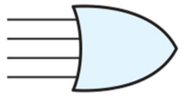
Concordia University
Department of Electrical & Computer Engineering

- To address 2^k memory locations (rows) we need a $k \times 2^k$ decoder.
- For example, a 4×4 RAM needs 2×4 decoder. Here, $k = 2$ and $2^k = 4$. Also, each word is 4 bits long.

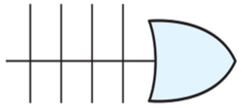


To save space:

instead of:



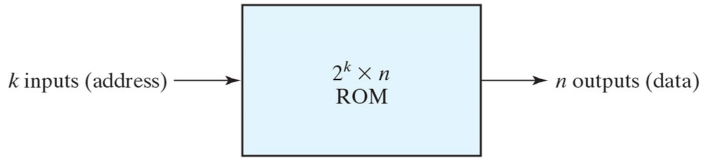
we use:



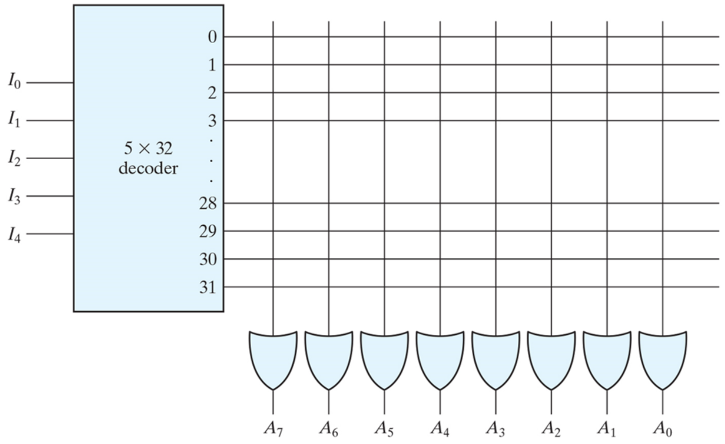
Lecture 12:
Read Only Memory (ROM):

Department of Electrical & Computer Engineering

- The block diagram of $2^k \times n$ ROM is:



- Address Decoding for a 32×8 ROM



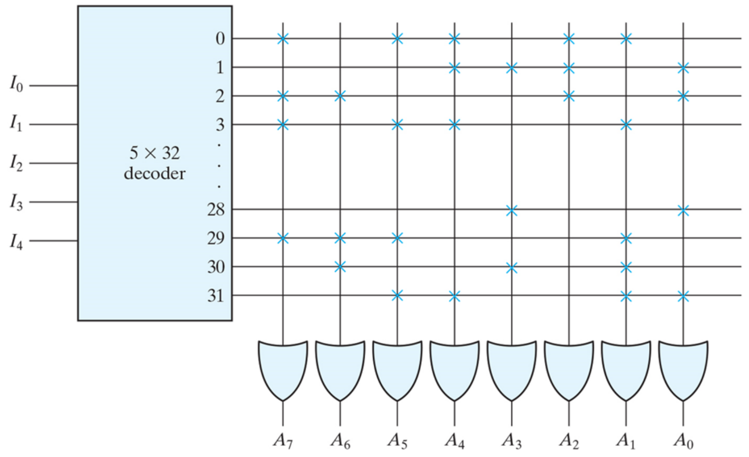
Lecture 12: Logic Design using ROM:

- Using a $2^k \times n$ ROM: n functions each with k variables can be designed.

Truth Table:

Address	Content A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0
00000	1	0	1	1	0	1	1	0
00001	0	0	0	1	1	1	0	1
00010	1	1	0	0	0	1	0	1
00011	1	0	1	1	0	0	1	0
⋮	⋮							
11100	0	0	0	0	1	0	0	1
11101	1	1	1	0	0	0	1	0
11110	0	1	0	0	1	0	1	0
11111	0	0	1	1	0	0	1	1

ROM Implementation:



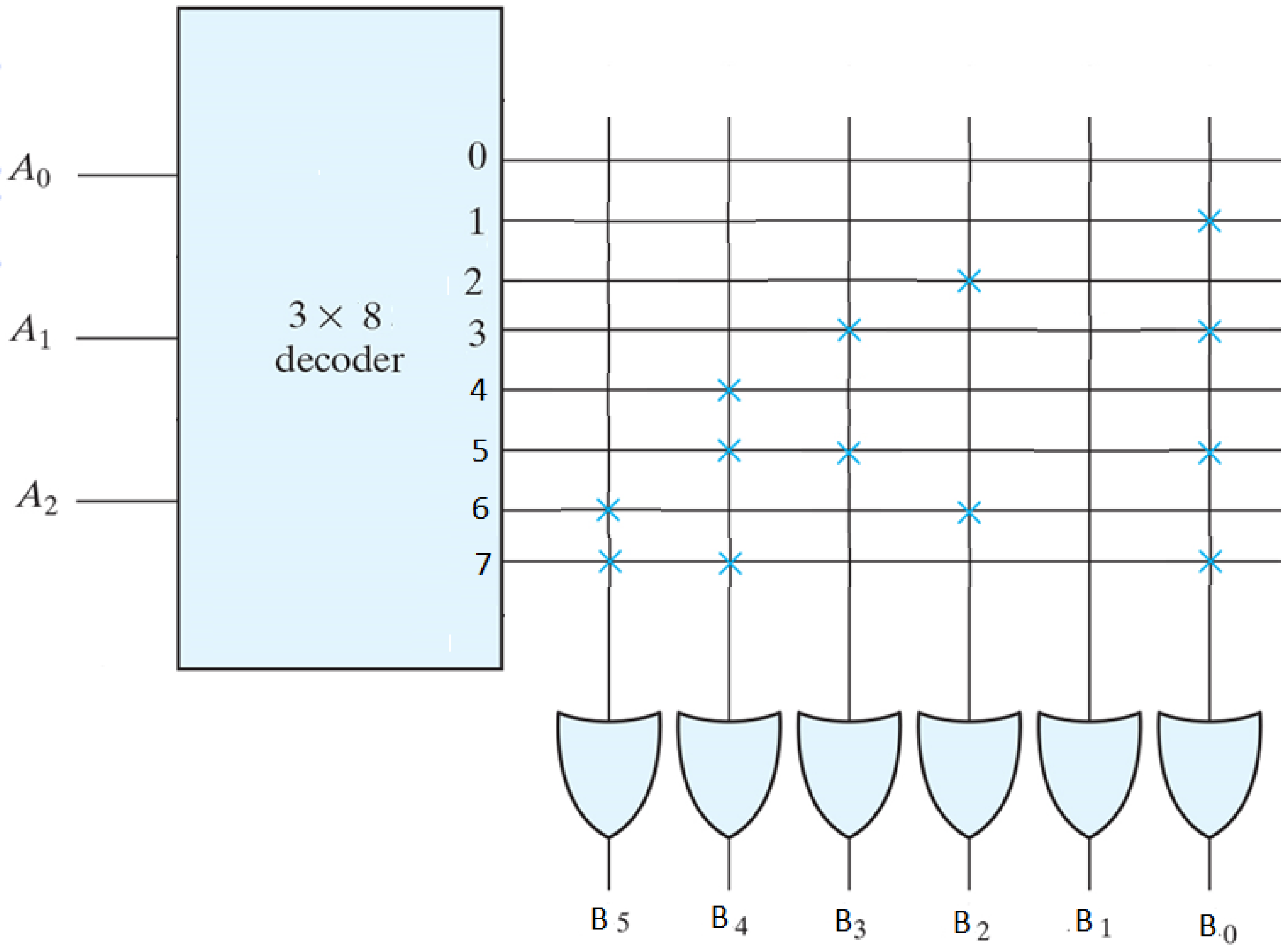
Each OR output is function, e.g., $A_7(I_4, I_3, I_2, I_1, I_0) = \sum(0,2,3,\ldots,29)$

Lecture 12:
Logic Design using ROM:

- **Example:** Design a circuit that has a 3-bit input and its output is the square of its inputs.
- The truth table is:

Inputs $A_2\ A_1\ A_0$			Outputs $B_5\ B_4\ B_3\ B_2\ B_1\ B_0$						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

Lecture 12: Logic Design using ROM:



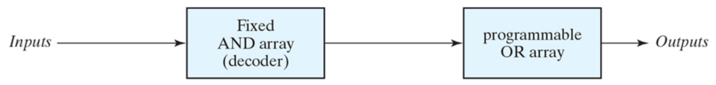
Lecture 12: Different Types of ROM:

- **Different Types of ROM:**
  - Mask Programmable ROM, or, simply ROM was programmed at the factory.
  - PROM: Programmable ROM could be programmed by the user using a PROM programmer, using high voltage pulses. The process is irreversible, i.e., once the data is written it cannot be erased.
  - EPROM: Erasable PROM. An EPROM can be erased using an ultraviolet source and programmed again.
  - EEPROM (E^2PROM): Electrically erasable PROM. This is similar to EPROM, but it can be erased using electric current.
  - Flash memory: It is similar to EEPROM, but it has internal circuitry enabling to write and erase in specific locations without needing a special programmer.

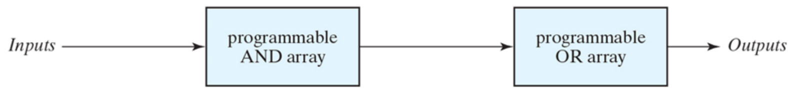
Lecture 12: Programmable Logic Devices (PLDs):

Department of Electrical & Computer Engineering

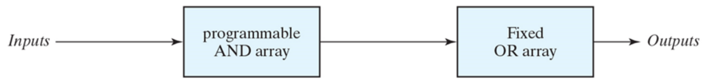
- There are three types of combinational PLDs:
  - PROM: in a PROM we have a set of fixed AND gates (forming the address decoder) and a set of programmable OR gates as we saw in previous examples.



  - Programmable Logic Array (PLA): In a PLA both AND gates and OR gates are programmable.



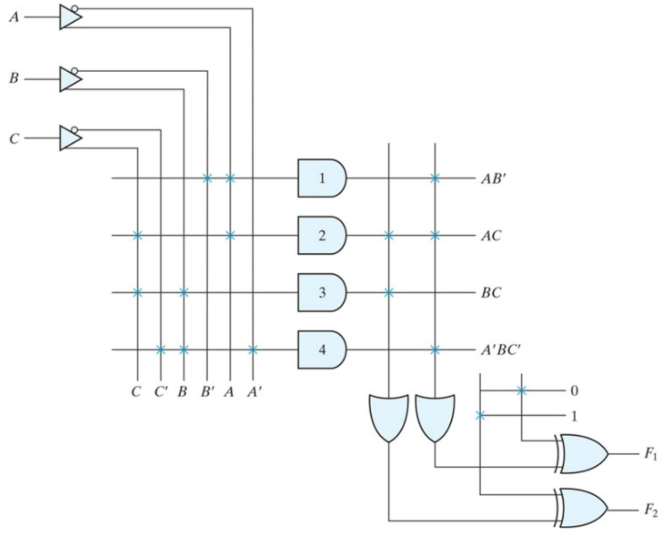
  - Programmable Array Logic (PAL): In a PAL the AND gates are programmable but the OR gates are fixed.



Lecture 12: Programmable Logic Array (PLA):

Department of Electrical & Computer Engineering

- A PLA is similar to PROM. The difference is that in PLA all the minterms are not generate. Only the terms required for implementing a function are formed.
- Following diagram shows a PLA with 3 inputs and two outputs.



The connection here implement:

$F_1 = AB' + AC + A'BC'$

and $F_2 = (AC + BC)$

The Programming Table:

		Inputs			Outputs (T)	Outputs (C)
	Product Term	***A***	***B***	***C***	F_1	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	—

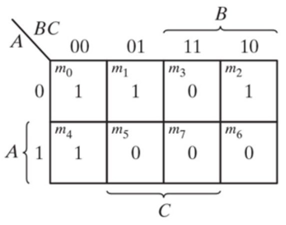
Lecture 12: Programmable Logic Array (PLA):

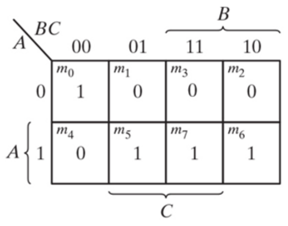
UNIVERSITÉ Concordia UNIVERSITY
Department of Electrical & Computer Engineering

- **Example:** Design the following function using the PLA in the previous example:

$$F_1(A,B,C) = \sum(0,1,2,4) \quad and \quad F_2(A,B,C) = \sum(0,5,6,7)$$

Solution: $F_1 = (AB + AC + BC)'$ and $F_2 = AB + AC + A'B'C'$





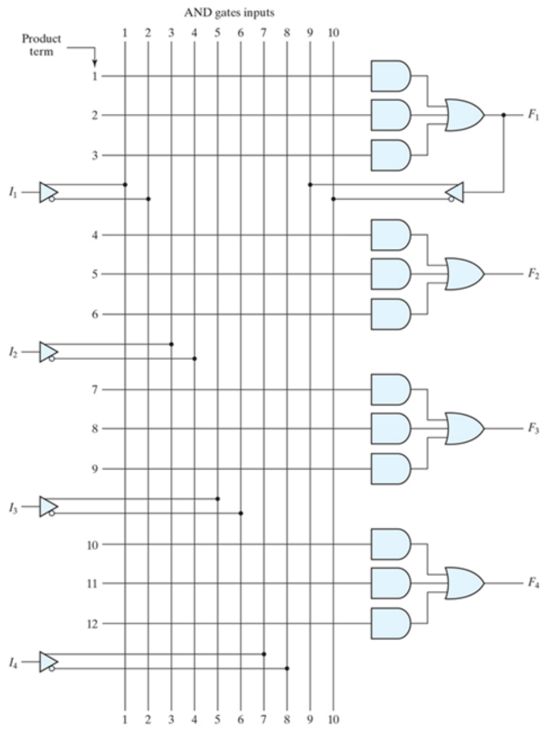
The fuse map is:

PLA programming table

	Product term	Inputs			Outputs (C)	Outputs (T)
		A	B	C	F_1	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

Lecture 12: Programmable Array Logic (PAL):

- PAL is a programmable device with fixed OR gates and programmable AND gates.
- This PAL has 4 sections each with
  3 programmable AND gates
  connected to a fixed OR gate.
  So, each function implemented can
  have at most 3 terms.
  If a function has more than 3 terms,
  we may implement it using two sections.



Lecture 12: Programmable Array Logic (PAL):

- Example: implement functions:

$w(A,B,C,D) = \sum(2,12,13)$, $x(A,B,C,D) = \sum(7,8,9,10,12,13,14,15)$
$y(A,B,C,D) = \sum(0,2,3,4,5,6,7,8,10,11,15)$ and $z(A,B,C,D) = \sum(1,2,8,12,13)$.

- Using K-map we find:

$$w = ABC' + A'B'CD'$$
$$x = A + BCD$$
$$y = A'B + CD + B'D'$$

and $z = ABC' + A'B'CD' + AC'D' + A'B'C'D$

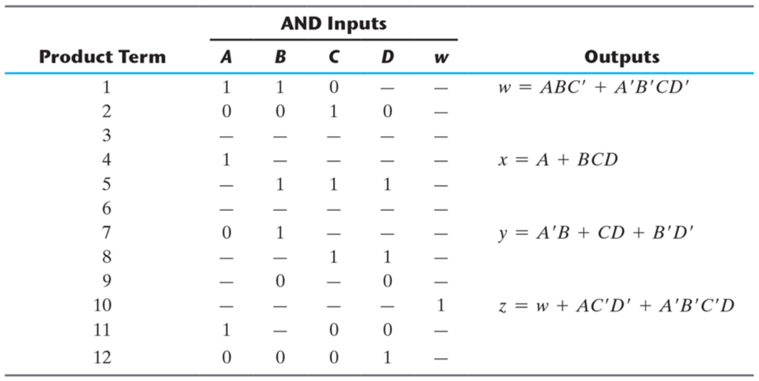
Lecture 12: Programmable Array Logic (PAL):

Concordia University
Department of Electrical & Computer Engineering

- z has more than 3 terms but since $ABC' + A'B'CD' = w$, we have:

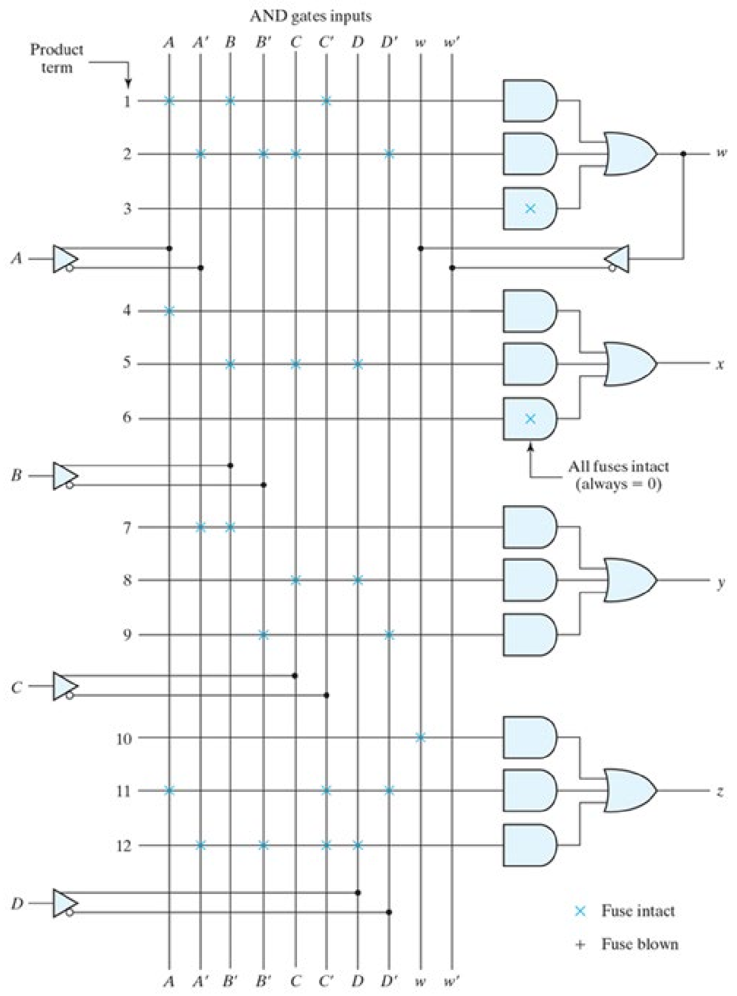
$$z = w + AC'D' + A'B'C'D$$

- Using w as one of the AND inputs we have:



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

Lecture 12: Programmable Array Logic (PAL):



Lecture 12: Knowledge Check

- **Question 1:** Assume you implement three functions each with 6 inputs using ROM. What should be the size of the ROM
- a) 1 k bytes, b) 1 kbits, c) 256 bits, d) 192 bits

- **Question 2:** In a PLA programmable gates are:
- a) AND, b) OR, c) both, d) buffer/inverter.