Lecture 17 (Chapter 3)

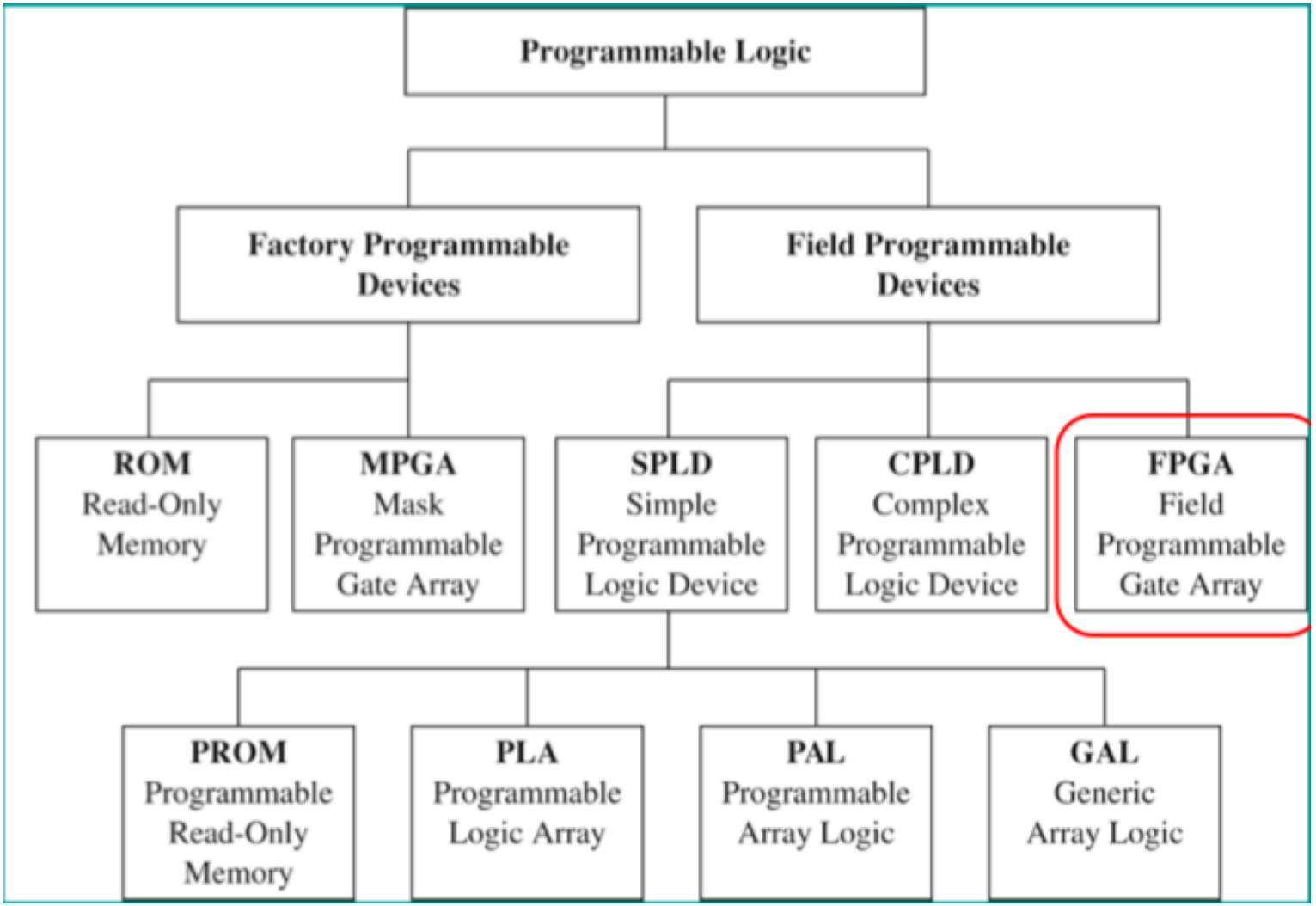
Programmable Logic Devices

Khaza Anuarul Hoque
ECE 4250/7250

Programmable Logic Devices

- Can be programmed by the users as per their requirement
- Large circuit is designed on a single chip
- Possible to implement a combinational or sequential circuit using the PLD
- Consist of array of NOT, AND and OR gates
- Any Boolean function can be represented by sum of the product form

Taxonomy


Programmable Logic
Factory Programmable Devices
Field Programmable Devices
ROM
Read-Only Memory
MPGA
Mask Programmable Gate Array
SPLD
Simple Programmable Logic Device
CPLD
Complex Programmable Logic Device
FPGA
Field Programmable Gate Array
PROM
Programmable Read-Only Memory
PLA
Programmable Logic Array
PAL
Programmable Array Logic
GAL
Generic Array Logic

SPLD/CPLD/FPGA

TABLE 3-1: A Comparison of Programmable Devices

	SPLD	CPLD	FPGA
Density	Low Few hundred gates	Low to Medium 500 to 12,000 gates	Medium to High 3,000 to 5,000,000 gates
Timing	Predictable	Predictable	Unpredictable
Cost	Low	Low to Medium	Medium to High
Major Vendors	Lattice Semiconductor Cypress AMD	Xilinx Altera	Xilinx Altera Lattice Semiconductor Actel
Example Device Families	**Lattice Semiconductor** GAL16LV8 GAL22V10 **Cypress** PALCE16V8 **AMD** 22V10	**Xilinx** CoolRunner XC9500 **Altera** MAX	**Xilinx** Virtex Spartan **Altera** Stratix **Lattice** Mach ECP **Actel** Accelerator

Programmable Array

Device	AND Array	OR Array
ROM	Fixed	Programmable
PLA	Programmable	Programmable
PAL	Programmable	Fixed
GAL	Programmable	Fixed

History of PLDs

- Programmable Logic Arrays ~ 1970
- Programmable Logic Devices ~ 1980
- Field Programmable Gate Arrays ~ 1985
- CPLD & FPGA architectures became similar ~ 2000

Programming technologies

- PLAs were mask programmable
- PALs used fuses for programming
- Early PLDs & CPLDs used floating gate technology
  - Erasable Programmable Read Only Memory (EPROM)
    - Ultra-violate erasable (UVEPROM)
    - Electrically erasable (EEPROM)
    - Flash memory came later and was used for CPLDs
- FPGAs used RAM for programming
- Later trends:
  - Fuses replaced by anti-fuse technology
  - Larger CPLDs went to RAM-based programming

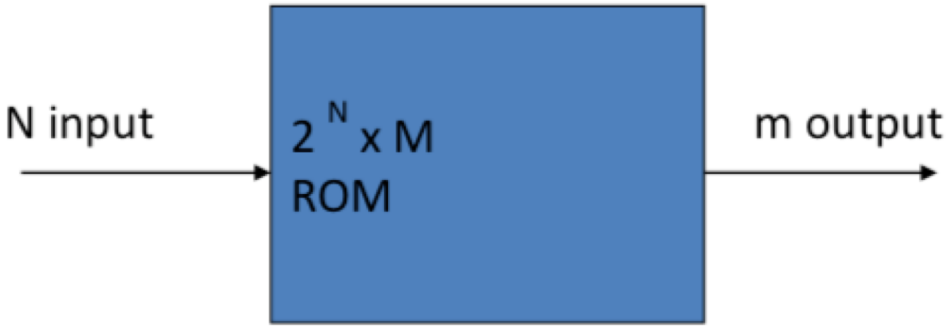
Read-Only Memory

- A read-only memory (ROM) consists of an array of semiconductor devices that are interconnected to store a set of binary data.
- Once binary data is stored in the ROM, it can be read out whenever desired, but the data that is stored cannot be changed under normal operating conditions.
- Data is written to the ROM once, and read from the ROM many times.

ROM Types

- **Programmable PROM**
  - Break links through current pulses
  - Write once, Read multiple times
- **Erasable PROM(EPROM)**
  - Program with ultraviolet light
  - Write multiple times, Read multiple times
- **Electrically Erasable PROM(EEPROM)/Flash Memory**
  - Program with electrical signal
  - Write multiple times, Read multiple times

Programmable ROM (PROM)



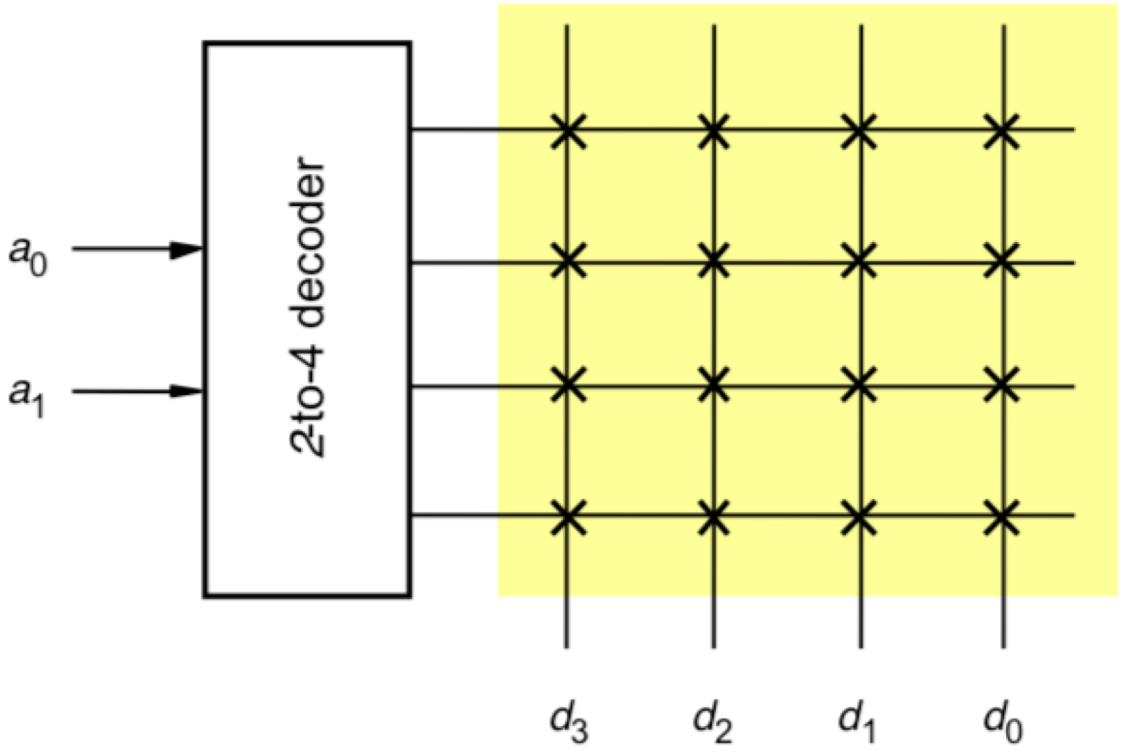
- Address: m bits; data: n bits
- ROM contains 2^N word of M bit each
- The input bits decide the particular word that becomes available on output lines

Programmable ROM (PROM)

- A ROM (Read Only Memory) has a fixed AND plane and a programmable OR plane
- Size of AND plane is 2^n where n = number of input pins
  - Has an AND gate for every possible minterm so that all input combinations access a different AND gate
- OR plane dictates function mapped by the ROM

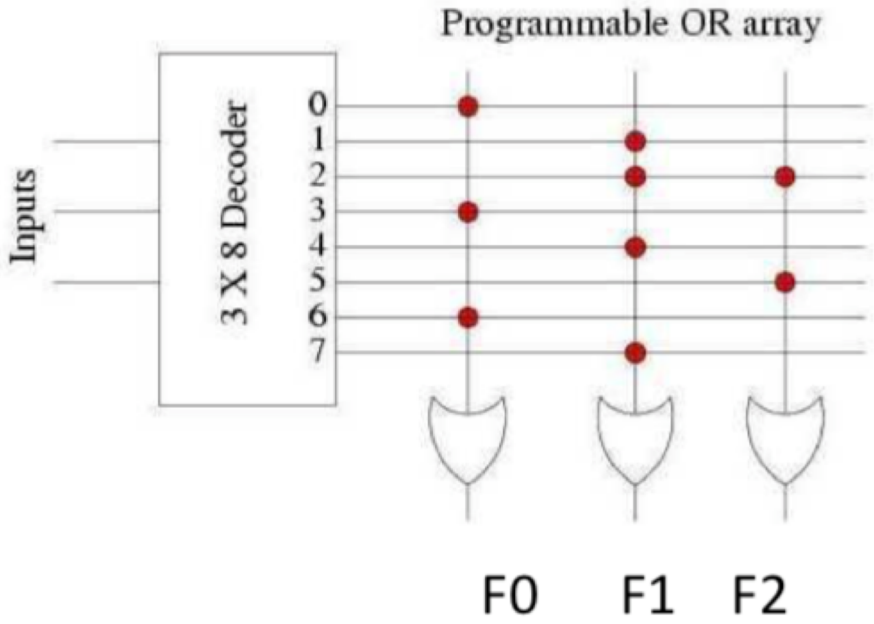
4x4 ROM

- 2^2x4 bit ROM has 4 addresses that are decoded

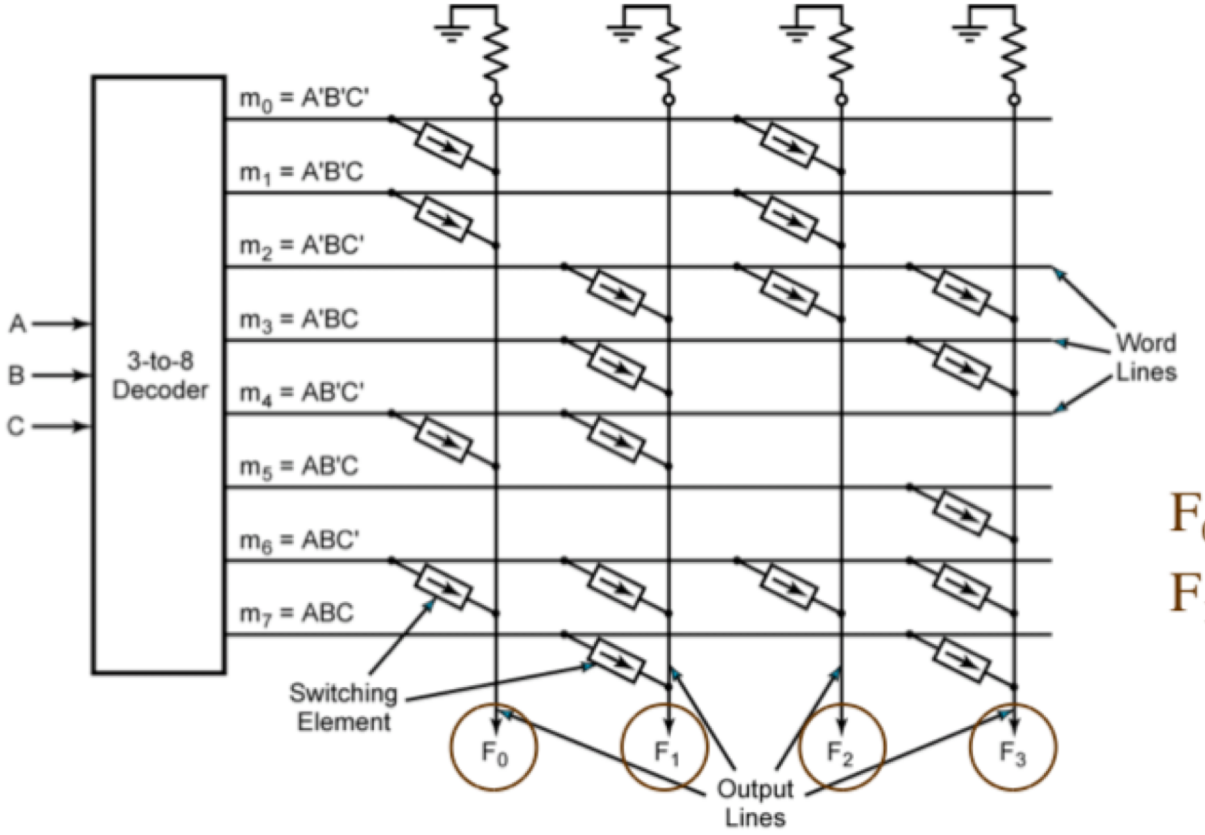


Combinational circuit implementation using ROM

I0	I1	I2	F0	F1	F2
0	0	0	1	0	0
0	0	1	0	1	0
0	1	0	0	1	1
0	1	1	1	0	0
1	0	0	0	1	0
1	0	1	0	0	1
1	1	0	1	0	0
1	1	1	0	1	0



Combinational circuit implementation using ROM



$F_0 = \Sigma m(0, 1, 4, 6)$

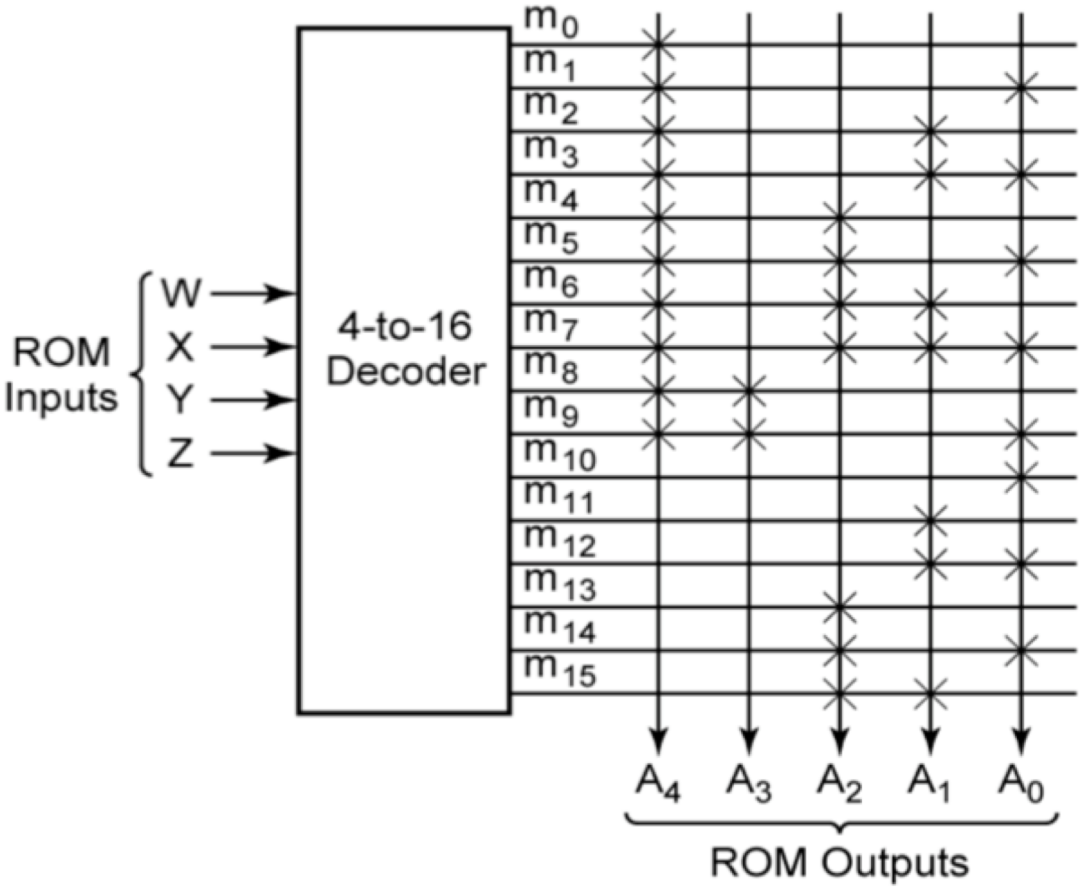
$F_1 = \Sigma m(2, 3, 4, 6, 7)$

What functions are realized by the ROM for F_2 and F_3?

HEX to ASCII

Input				Hex	ASCII Code for Hex Digit						
W	X	Y	Z	Digit	A_6	A_5	A_4	A_3	A_2	A_1	A_0
0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	1	1	0	1	1	0	0	0	1
0	0	1	0	2	0	1	1	0	0	1	0
0	0	1	1	3	0	1	1	0	0	1	1
0	1	0	0	4	0	1	1	0	1	0	0
0	1	0	1	5	0	1	1	0	1	0	1
0	1	1	0	6	0	1	1	0	1	1	0
0	1	1	1	7	0	1	1	0	1	1	1
1	0	0	0	8	0	1	1	1	0	0	0
1	0	0	1	9	0	1	1	1	0	0	1
1	0	1	0	A	1	0	0	0	0	0	1
1	0	1	1	B	1	0	0	0	0	1	0
1	1	0	0	C	1	0	0	0	0	1	1
1	1	0	1	D	1	0	0	0	1	0	0
1	1	1	0	E	1	0	0	0	1	0	1
1	1	1	1	F	1	0	0	0	1	1	0

HEX to ASCII



$A_5 = A_4$

$A_6 = A_5'$