Read-Only Memory

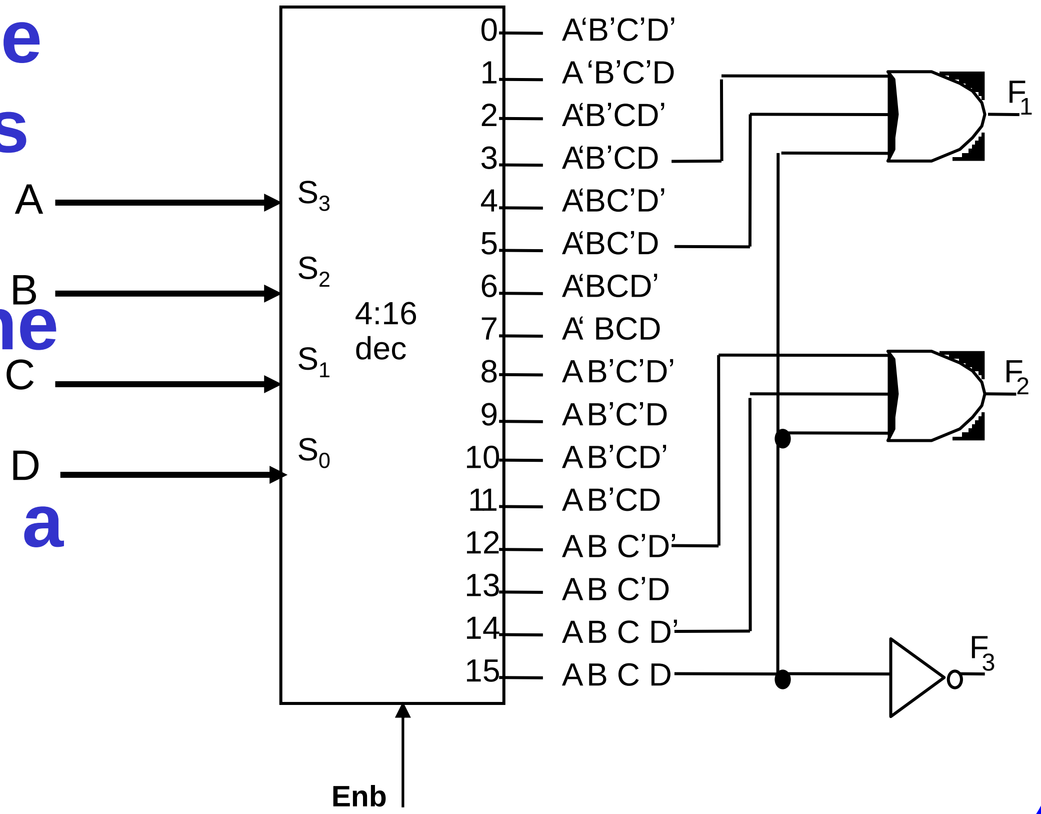
ROM

ROM

- Can be used:
  - to implement any arbitrary combinational circuit
  - as a memory
- Consists of:
  - an n-to-2^n decoder that produces ALL minterms
  - a set of programmable OR gates that produce SoP's
- Is usually described in terms of:
  - size of its decoder output (number of memory rows)
  - number of OR gates (memory width)
  - i.e., 2^n x w, e.g., 8x4, 1024x8, etc.

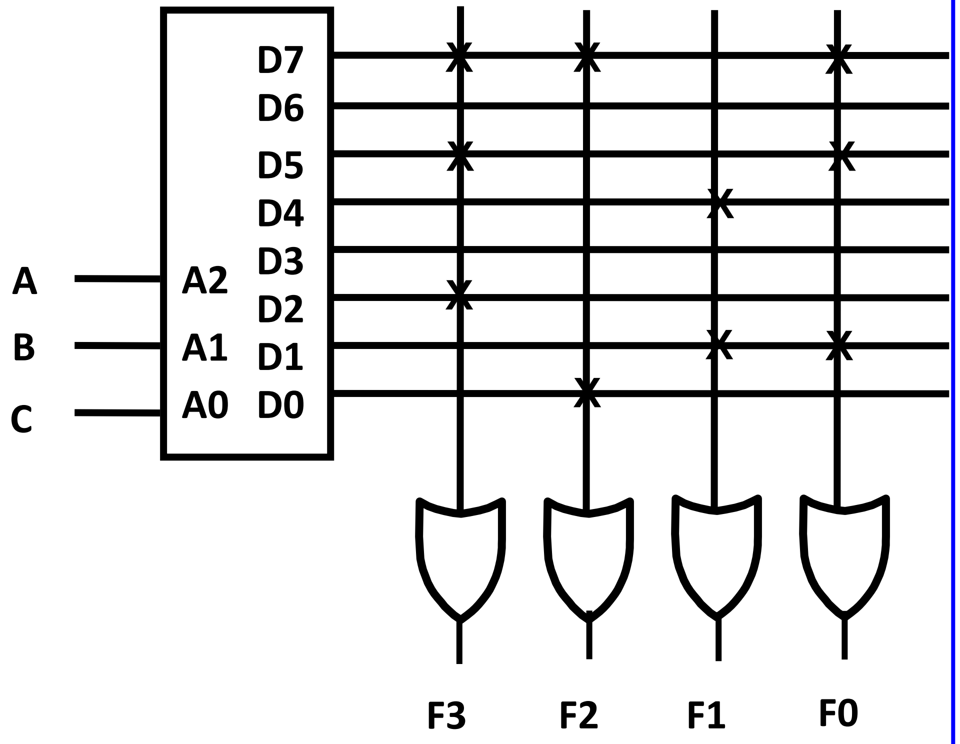
ROM: Example 1

- A 4-to-16 decoder
- Three OR gates
- Implemented three Boolean functions
- Has an "enable" input to control the output
- Can be viewed as a 16 x 3 memory
- Memory content?

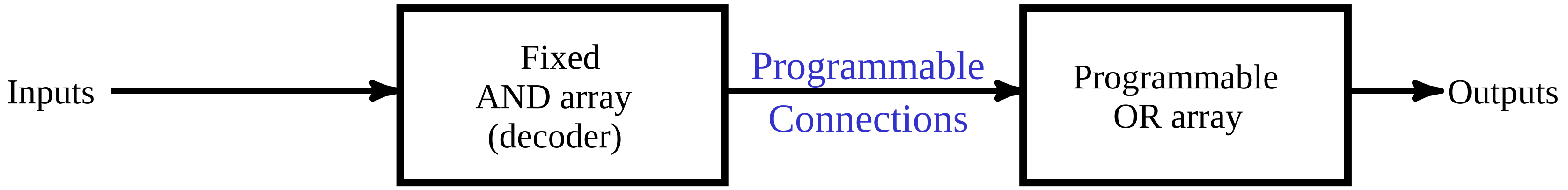


ROM: Example 2

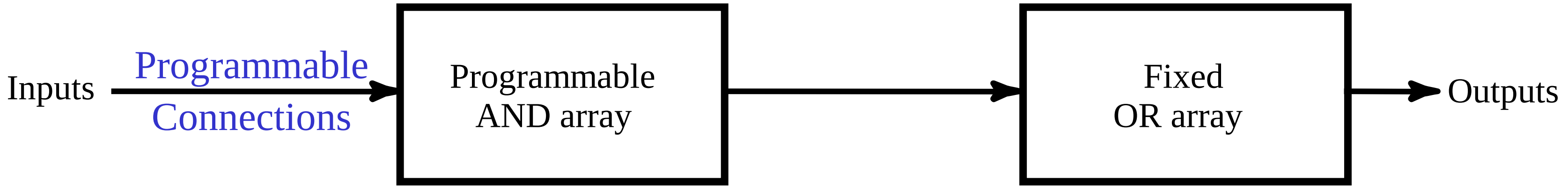
- A 3-to-8 decoder
- Four OR gates
- Can implement four Boolean functions
- Can be viewed as a 8 x 4 memory
- No “enable” input
- Memory content?



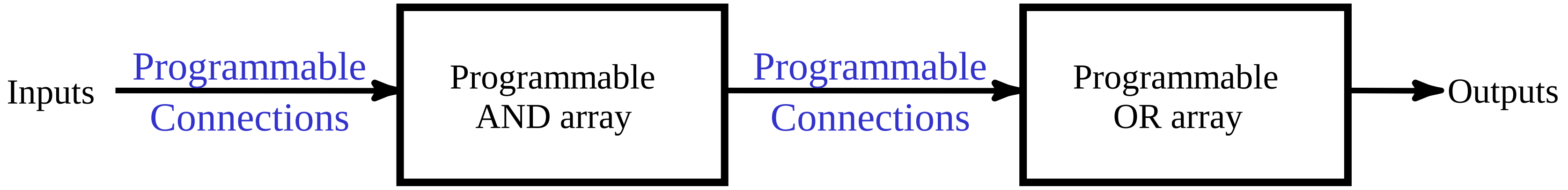
ROM vs. PLA/PAL



(a) Programmable read-only memory (PROM)



(b) Programmable array logic (PAL) device



(c) Programmable logic array (PLA) device

General Logic Implementation

- ➢ Given a 2^kxn ROM, we can implement ANY combinational circuit with <u>at most</u> k inputs and <u>at most</u> n outputs.

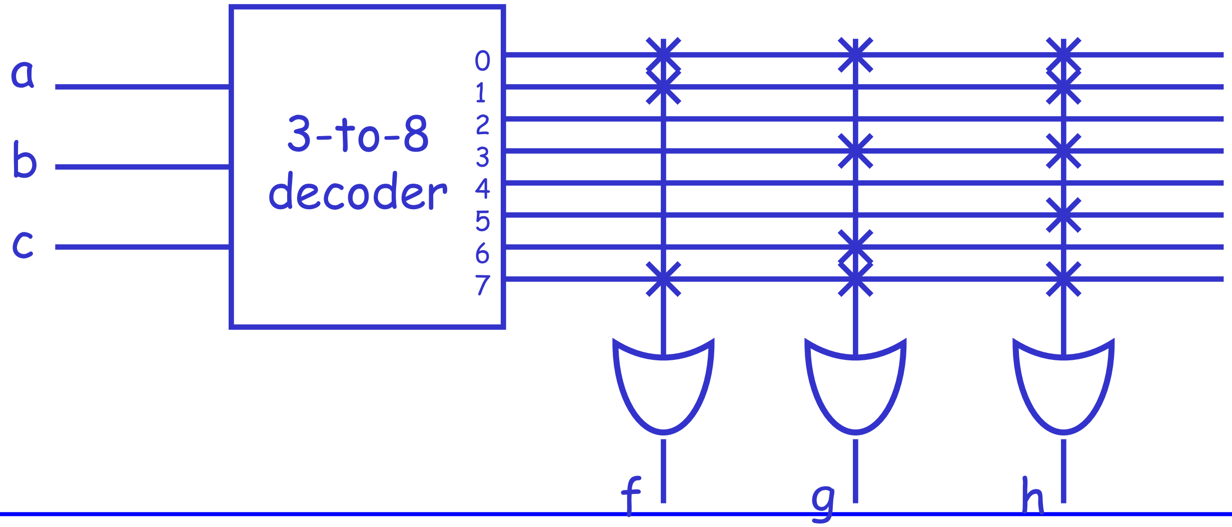
- **Why?**
  - ➢ k-to-2^k decoder will generate all 2^k possible minterms
  - ➢ Each of the OR gates can implement a $\sum m()$
  - ➢ Each $\sum m()$ can be programmed to represent one function

Example

- **Find a ROM-based circuit implementation for:**
  - f(a,b,c) = a'b' + abc
  - g(a,b,c) = a'b'c' + ab + bc
  - h(a,b,c) = a'b' + c
- **Solution:**
  1. Determine the required ROM size
  2. Express f(), g(), and h() in $\sum$m() format (use truth tables)
  3. Program the ROM, based on the 3 $\sum$m()'s

Example (Continued)

1. **There are 3 inputs and 3 outputs, thus we need an 8x3 ROM block.**
2. **Prepare the minterm lists:**
   - $f = \sum m(0, 1, 7)$
   - $g = \sum m(0, 3, 6, 7)$
   - $h = \sum m(0, 1, 3, 5, 7)$
3. **Program the ROM**

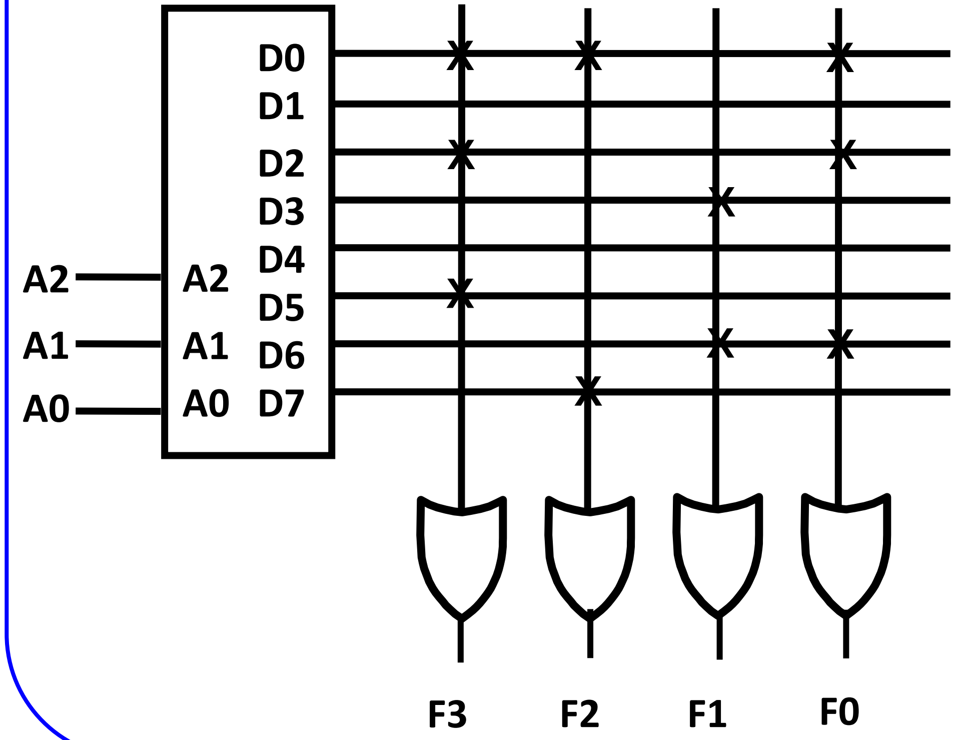


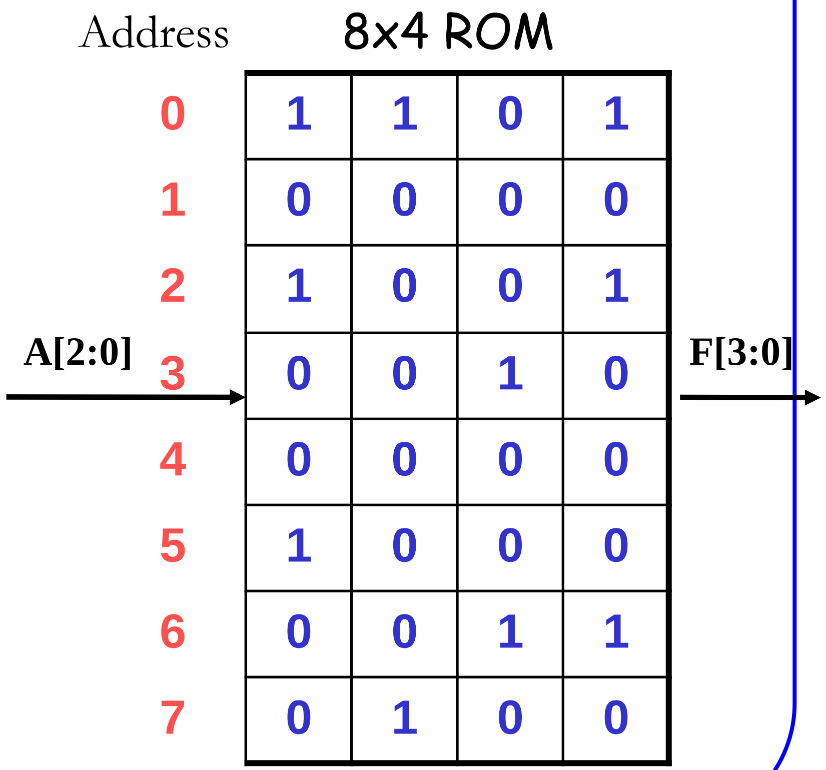
ROM as a Memory

- **ROM's can be viewed as memory with the inputs as address lines, and outputs as the stored data**
- **Usually have:**
    - N input lines,
    - M output lines,
    - Provide 2^N x M bits of memory

ROM as Memory (Example)

- Read Example: For input $(A_2,A_1,A_0) = 011$, output is $(F_3,F_2,F_1,F_0) = 0010$.
- What are functions F_3, F_2, F_1 and F_0 in terms of (A_2, A_1, A_0)?





Address	8x4 ROM			
0	1	1	0	1
1	0	0	0	0
2	1	0	0	1
3	0	0	1	0
4	0	0	0	0
5	1	0	0	0
6	0	0	1	1
7	0	1	0	0

ROM as a Memory

- **History:**
  - ROM: the first generation, used as a memory but preprogrammed at the time of manufacturing
  - PROM: Programmable ROM, the second generation, able to be programmed at the time of usage
  - EPROM: Erasable PROM, able to be erased by UV, and reprogrammed
  - EEPROM: Electronically EPROM, able to be erased electronically and reprogrammed

(Memories)

- **Volatile:**
  - Random Access Memory (RAM):
    - SRAM: "static"
    - DRAM: "dynamic"

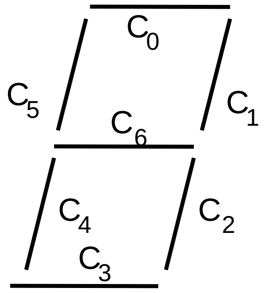
- **Non-Volatile:**
  - ROM
  - PROM
  - EPROM
  - EEPROM
  - FLASH memory: similar to EEPROM with programmer integrated on chip

Design by ROM: Example

- **BCD to 7-Segment Display Controller (Decoder)**

A B C D	C0	C1	C2	C3	C4	C5	C6
0 0 0 0	1	1	1	1	1	1	0
0 0 0 1	0	1	1	0	0	0	0
0 0 1 0	1	1	0	1	1	0	1
0 0 1 1	1	1	1	1	0	0	1
0 1 0 0	0	1	1	0	0	1	1
0 1 0 1	1	0	1	1	0	1	1
0 1 1 0	1	0	1	1	1	1	1
0 1 1 1	1	1	1	0	0	0	0
1 0 0 0	1	1	1	1	1	1	1
1 0 0 1	1	1	1	0	0	1	1
1 0 1 0	X	X	X	X	X	X	X
1 0 1 1	X	X	X	X	X	X	X
1 1 0 0	X	X	X	X	X	X	X
1 1 0 1	X	X	X	X	X	X	X
1 1 1 0	X	X	X	X	X	X	X
0 1 1 1	X	X	X	X	X	X	X

Reminder:

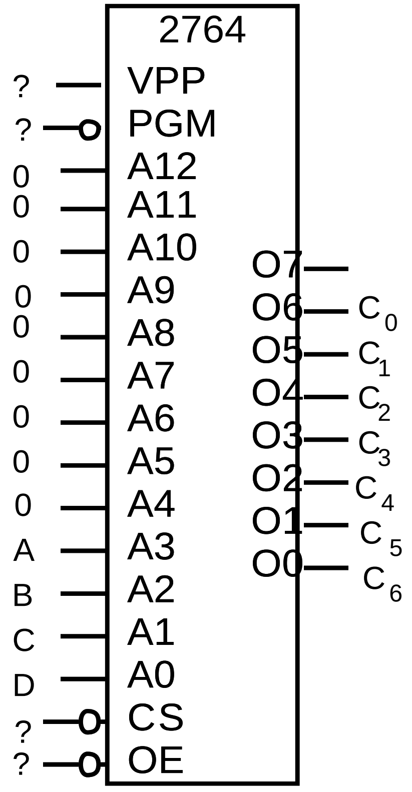


Need a ROM with at least 4 address lines and 7 bits of output

Design by ROM: Example Continued

- There are some standard devices such as 2764
- Vpp and PGM are used when programming
- Chip Select (CS) and Output Enable (OE) inputs are used to control the chip
- 13 address lines provide 2^{13} (=8192=8K) memory bytes of 8 bits
- Extra address lines grounded
- Extra output line not connected

2764 EPROM
8K x 8



ROM vs. PLA/PAL

- **ROM approach advantageous when**
  - (1) design time is short (no need to minimize output functions)
  - (2) little sharing of product terms among output functions

- **ROM problem:**
  - (1) Size doubles for each additional input,
  - (2) Can't use don't cares

- **PLA approach advantageous when**
  - (1) a design tool (like espresso) is available
  - (2) many minterms are shared among the output functions

- **PLA problem:**
  - ➢ Too much programmability for ordinary applications => too expensive

- **PAL problem:**
  - ➢ Constrained fan-ins on OR planes

ROM vs. PLA

- **ROM**
  - Cheap (high-volume component)
  - Medium speed

- **PLA**
  - Expensive
    - Complex design; need more sophisticated tools
  - Slow
    - Two programmable planes