Programmable Logic Devices

- Read Only Memories (ROMs)
- Programmable Logic Arrays (PLAs)
- Programmable Array Logic Devices (PALs)
3.1 Read-Only Memories

- Store binary data
- \( n \) input lines, \( m \) output lines => array of \( 2^n m \)-bit words
- Use ROM to implement logic functions?

3.1 Read-Only Memories - Basic ROM Structure

- Decoder
- Memory Array
- ROM
- \( n \) input lines
- \( m \) output lines
- \( 2^n \) words \( \times \) \( m \) bits
3.1 Read-Only Memories - ROM Types

- Mask-programmable ROM
  - 
  - 
- EPROM (Erasable Programmable ROM)
  - 
  - 
  - EEPROM — Electrically Erasable PROM
- Flash memories - similar to EEPROM except they use a different charge-storage mechanism
  - Usually have built-in programming and erase capability, so the data can be written to the flash memory while it is in place, without the need for a separate programmer

3.2 Programmable Logic Arrays (PLAs)

- Perform the same function as a ROM
  - \( n \) inputs and \( m \) outputs — 
  - AND array — 
  - OR array — 

![Diagram of a Programmable Logic Array (PLA)]
3.2 Programmable Logic Arrays - Example

\[ \sum_{i=0}^{2} m(0, 4, 6) = A'B' + AC' \]
3.2 Programmable Logic Arrays - AND-OR Array Equivalent

Modified Truth Table for PLA

\[ F_0 = \Sigma m(0, 1, 4, 6) = A'B' + AC' \]
\[ F_1 = \Sigma m(2, 3, 4, 6, 7) = B + AC' \]
\[ F_2 = \Sigma m(0, 1, 2, 6) = A'B' + BC' \]
\[ F_3 = \Sigma m(2, 3, 5, 6, 7) = AC + B \]

<table>
<thead>
<tr>
<th>Product Term</th>
<th>Inputs</th>
<th>F0</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>A B</td>
<td>0 0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>AC</td>
<td>1 -</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>0 1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>BC</td>
<td>- 1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>AC</td>
<td>1 -</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
Using PLA: An Example

\[ F_1 = \sum m(2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 15) \]
\[ F_2 = \sum m(2, 3, 5, 6, 7, 10, 11, 14, 15) \]
\[ F_3 = \sum m(6, 7, 8, 9, 13, 14, 15) \]
\[ F_1 = bd + b'c + ab' \]
\[ F_2 = c + a'bd \]
\[ F_3 = bc + ab'c' + abd \]

Eight different product terms are required!?

For PLA we want to minimize the total number of product terms, not the number of product terms for each function separately!

3.2 Programmable Logic Arrays - How Many Product Terms are Needed?
### Using PLA: An Example

- **F_1** = \( a'b'd + ab'd + ab'c' + b'c \)
- **F_2** = \( a'bd + b'c + bc \)
- **F_3** = \( abd + ab'c' + bc \)

### 3.3 Programmable Array Logic (PALs)

- **PAL** is a special case of PLA
  - AND array is _______________ and OR array is __________
- **PAL** is
  - less expensive
  - easier to program
3.3 Programmable Array Logic (PALs)

Unprogrammed

Programmed

Typical PALs have
- from 10 to 20 inputs
- from 2 to 10 outputs
- from 2 to 8 AND gates driving each OR gate
- often include D flip-flops
3.3 Programmable Array Logic - Logic

Diagram for 16R4 PAL - Top Half