Chapter 6 – Part 2
Memories & Programmable Logic Devices

Overview of Chapter 6

Part 1: Memories
- Memory Definitions
- Random Access Memory
  - Function, Operation, Timing
- RAM Integrated Circuits
  - RAM Cell, RAM Bit Slice
  - Static Buffer
  - Cell Arrays and Coincident Selection
  - Dynamic RAM
- Array of RAM Integrated Circuits
  - Making larger and wider memories

Part 2: Programmable Logic Technologies
- Introduction
- Read-Only Memory
- Programmable Logic Array (PLA)
- Programmable Array Logic (PAL)
- Examples and implementations

Programmable Logic

- IC Cost Dilemma:
  - IC logic circuit density increases exponentially with time.
  - The more ICs you make, the cheaper they get.
  - Complex logic ICs have very specific functions (so you make fewer).

- Question: How do I make very high volume parts that are very dense?
  - Answer #1: You make microprocessors or ICs (like automobile ignition controllers) that have very large volume. OR
  - Answer #2: You make programmable parts.

Programmable Part Types

- Semiconductor manufacturers have developed several types of regular programmable logic elements. Three important ones are:
  - Read Only Memory (ROM) – a fixed array of AND gates and a programmable array of OR gates.
  - Programmable Array Logic (PAL) – a programmable array of AND gates feeding a fixed array of OR gates.
  - Programmable Logic Array (PLA) – a programmable array of AND gates feeding a programmable array of OR gates.

- All of the above use regular structures of logic elements that can be thought of as a “memory array”. There are also programmable devices that look more like programmable logic cells with programmable interconnect.

Programming Devices

- Devices may be:
  - Permanently programmed at the time of IC manufacture,
  - Programmed at the time of use (board level manufacturing), or
  - Dynamically re-programmed during use.

- Permanent programming techniques done at the time of manufacture include final level interconnect addition via metallization or device alteration through laser or e-beam programming.

- Use time programming techniques include shorting diodes, blowing fuses, shorting devices, and dumping charge into wells.

- Dynamically reprogrammed devices can be bulk erased and reprogrammed, or incrementally erased and reprogrammed.

Read Only Memory

- Read Only Memories (ROM) or Programmable Read Only Memories (PROM) have:
  - N input lines,
  - M output lines, and
  - 2^N decoded minterms.

- The N input lines are connected to a fixed decoder AND array of 2^N lines. Each line represents a minterm of N variables. Thus there are 2^N decoded minterms.

- Each of the M output lines is connected to an OR gate which has a programmable number of input connections. Any (or all) of the minterms may be ORed together for each of the M output lines.

- A program map for a PROM (or ROM) LOOKS LIKE A MULTIPLE OUTPUT FUNCTION TABLE.
Read Only Memories (Continued)

- Example: A 8 X 4 ROM
  \( (N = 3 \text{ input lines}, M = 4 \text{ output lines}) \)
- The fixed "AND" array is a decoder with 3 input bits to one-of-8 minterm output lines.
- The programmable "OR" array is shown as a "Wire-OR" function. A "Dot" in the array corresponds to including that minterm.

See the diagram for a visual representation of the fixed and programmable arrays.

Read Only Memories (Continued)

- The 32 X 8 ROM example corresponds to the multiple output truth table:

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>A B C</td>
<td>F0 F1 F2 F3</td>
</tr>
<tr>
<td>0 0 0</td>
<td>1 1 0 1</td>
</tr>
<tr>
<td>0 0 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>0 1 0</td>
<td>0 0 0 1</td>
</tr>
<tr>
<td>0 1 1</td>
<td>0 0 0 0</td>
</tr>
<tr>
<td>1 0 0</td>
<td>0 0 1 0</td>
</tr>
<tr>
<td>1 0 1</td>
<td>1 0 1 0</td>
</tr>
<tr>
<td>1 1 0</td>
<td>0 0 0 1</td>
</tr>
<tr>
<td>1 1 1</td>
<td>0 1 0 0</td>
</tr>
</tbody>
</table>

Programmable Array Logic (PAL)

- PAL devices are closely related to the ROM in that the device is organized as a regular array of programmable elements.
- The PAL has a programmable set of AND terms, combined with a limited number of fixed OR terms.
- Where the ROM array is guaranteed to implement any function of N inputs, the PAL may run out of OR terms. Thus, it may be very important to minimize the number of OR terms in order to use a PAL.
- Another difference is that a ROM does not easily allow multi-level implementations. The designer must use separate ROMs for multiple levels. The PAL allows outputs from OR terms to be used as inputs to AND terms, making multi-level design easy.

Programmable Array Logic (Cont.)

- An "X" at a cross line includes that variable in the AND term. An "X" in an AND gate removes that term. An "X" at the EXOR forms a "TRUE" term, else the output is complemented. Thus we have:

\[
\text{Out1} = (\text{In1} \oplus \text{In2} \oplus \text{In3}) + \text{In4}
\]

- Note that Out1 leaves the PAL in COMPLEMENT form, since the EXOR is tied to one. Out2 is shown in "TRUE" form.

Programmable Array Logic (Cont.)

- Example: 4 Input, 3 Output PAL with fixed, 3-input OR terms and programmable polarity outputs.
- This device is unprogrammed.

Programmable Array Logic (Cont.)

- What are the equations for the other terms?
Programmable Logic Array (PLA)

- The last type of programmable logic element we will discuss is the PLA which has a programmable array of AND and OR terms.
- A PLA typically has a large number of inputs and outputs and can be used to implement equations that are impractical for a ROM (because of the number of inputs required).
- Generally the product terms limit the application of a PLA. Use minimization techniques to reduce the number of product terms in an implementation if it is to fit in a PLA.
- The program for a PLA is very similar to the connection array for a multiple output Boolean function, such as that generated by CAFE.

-ROM, PAL, PLA Examples

- RECALL: Square Root of a Number (INT)
  - INT(0)
    - y
    - x
    - z
  - INT(1)
    - 1 1 1 1
    - 1 1 1 1
    - 1 1 1 1
    - 1 1 1 1

Summary of Programmable Logic

- Device ⇒ Characteristics
- Device Uses

<table>
<thead>
<tr>
<th>Type</th>
<th>AND terms</th>
<th>OR terms</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROM</td>
<td>2^n Fixed terms</td>
<td>Programmable</td>
</tr>
<tr>
<td>PAL</td>
<td>Programmable</td>
<td>Fixed</td>
</tr>
<tr>
<td>PLA</td>
<td>Programmable</td>
<td>Programmable</td>
</tr>
</tbody>
</table>

ROM, PAL, PLA Examples

- Function Table (for reference)
- NUM INT
- 0000 10
- 0001 01
- 0010 01
- 0011 01
- 0100 10
- 0101 10
- 0110 10
- 0111 11

ROM Implementation

- This implementation is trivial – the function table serves as the fuse map. A 16-word by 2-bit prom is needed (we will use a 16x4 PROM as shown):
- Try programming it yourself first.
For the PAL implementation, we go back to the minimized equations:

\[
\begin{align*}
\text{INT}(1) &= w + x \\
\text{INT}(0) &= \overline{x} + \overline{y} + \overline{z} + wx
\end{align*}
\]

Try your hand at programming it!

Here's an implementation:

\[
\begin{align*}
\text{INT}(1) &= w + x \\
\text{INT}(0) &= \overline{x} + \overline{y} + \overline{z} + wx
\end{align*}
\]

Is this correct?

For the PAL implementation, we also use the minimized equations:

\[
\begin{align*}
\text{INT}(1) &= w + x \\
\text{INT}(0) &= \overline{x} + \overline{y} + \overline{z} + wx
\end{align*}
\]

Try your hand at programming it!

Here is an implementation:

\[
\begin{align*}
\text{INT}(1) &= w + x \\
\text{INT}(0) &= \overline{x} + \overline{y} + \overline{z} + wx
\end{align*}
\]

Is it correct?