Chapter 7. Memory and Programmable Logic
7.1 Introduction

• Memory unit:
  • A device to which binary information is transferred for storage and from which information is retrieved when needed for processing
  • A collection of cells capable of storing a large quantity of binary information

• RAM
  • Random access memory
  • Write/read operations

• ROM
  • Read only memory
  • Programmable logic device (PLD), programmable logic array (PLA), programmable array logic (PAL), field-programmable gate array (FPGA)

FIGURE 7.1 Conventional and array logic diagrams for OR gate
7.2 Random-Access Memory

- Time to transfer information to or from any desired random location is same

- Words:
  - Binary information in groups of bits
  - Byte: A group of 8 bits
  - 16-bit word, 32-bit word

- Data input and output lines
- Address selection lines
  - Identification number for selecting one particular word
- Control lines specify the direction of transfer
- $K(kilo=2^{10})$, $M(mega=2^{20})$, $G(giga=2^{30})$
## FIGURE 7.3 Contents of a 1024 × 16 memory

<table>
<thead>
<tr>
<th>Binary</th>
<th>Decimal</th>
<th>Memory content</th>
</tr>
</thead>
<tbody>
<tr>
<td>000000000000</td>
<td>0</td>
<td>10110101010111101</td>
</tr>
<tr>
<td>000000000001</td>
<td>1</td>
<td>1010101110001001</td>
</tr>
<tr>
<td>00000000010</td>
<td>2</td>
<td>0000110101000110</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1111111101</td>
<td>1021</td>
<td>1001110100010100</td>
</tr>
<tr>
<td>1111111110</td>
<td>1022</td>
<td>0000110100011110</td>
</tr>
<tr>
<td>1111111111</td>
<td>1023</td>
<td>1101111000100101</td>
</tr>
</tbody>
</table>
Write and Read Operations

• Write operation
  1. Apply the binary address of the desired word to the address lines
  2. Apply the data bits that must be stored in memory to the data input lines
  3. Activate the write input

• Read operation
  1. Apply the binary address of the desired word to the address lines
  2. Activate the read input

Table 7.1
Control Inputs to Memory Chip

<table>
<thead>
<tr>
<th>Memory Enable</th>
<th>Read/Write</th>
<th>Memory Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>X</td>
<td>None</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Write to selected word</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Read from selected word</td>
</tr>
</tbody>
</table>
Timing Waveforms

- **Access time:**
  - Time required to select a word and read it

- **Cycle time:**
  - Time required to complete a write operation

- **Example**
  - CPU 50MHz clock (20ns)
  - 50 ns maximum cycle time
Types of Memories

- Access time
  - Random-access memory: same access time
  - Sequential-access memory: magnetic disc or tape, variable access time

- Operating modes
  - Static RAM (SRAM): internal latches
  - Dynamic RAM (DRAM): MOS transistors for storing binary information as electric charges on the capacitors; require refreshing the dynamic memory (recharge)

- Volatile
  - Loss stored information when power down (CMOS integrated circuit RAM; SRAM, DRAM)

- Nonvolatile
  - Retain the stored information after the removal of power (Magnetic disk)
7.3 Memory Decoding

- RAM of \( m \) words and \( n \) bits per word: \( m \times n \) binary storage cells
- Associated decoding circuits for selecting individual words
- SR latch
- To pack as many cells as possible in the small area

**FIGURE 7.5 Memory cell**
Logical Construction of RAM

- 4 words of 4 bits each
- $2^k$ words of $n$ bits per word
- Require $k$ address lines and $k \times 2^k$ decoder

**FIGURE 7.6** Diagram of a $4 \times 4$ RAM
Coincident Decoding

- Two $k/2$-input decoders used instead of $k$-input decoder
- Selected by the coincidence of one X line and one Y line
- Each intersection represents a word

**FIGURE 7.7** Two-dimensional decoding structure for a 1K-word memory
Address Multiplexing

- Large capacity
- Reduce the size
- Row address strobe (RAS)
- Column address strobe (CAS)

**FIGURE 7.8** Address multiplexing for a 64K DRAM
7.4 Error Detection and Correction

- Occasional errors in storing and retrieving the binary information
- Employing error-detecting and error-correcting codes
- Parity bit check (detect, but cannot correct)
- Syndrome: generate a unique pattern
- Hamming code
  - Add $K$ parity bits to an $n$-bit data word
  - Positions numbers as a power of 2 for parity bits

\[
\begin{array}{cccccccccccc}
\text{Bit position:} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \\
\quad P_1 & P_2 & 1 & P_4 & 1 & 0 & 0 & P_8 & 0 & 1 & 0 & 0 \\
\end{array}
\]

\[
P_1 = \text{XOR of bits } (3, 5, 7, 11) = 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 = 0 \\
P_2 = \text{XOR of bits } (3, 5, 7, 10, 11) = 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 = 0 \\
P_4 = \text{XOR of bits } (5, 6, 7, 12) = 1 \oplus 0 \oplus 0 \oplus 0 = 1 \\
P_8 = \text{XOR of bits } (9, 10, 11, 12) = 0 \oplus 1 \oplus 0 \oplus 0 = 1 \\
C_1 = \text{XOR of bits } (1, 3, 5, 7, 9, 11) \\
C_2 = \text{XOR of bits } (2, 3, 6, 7, 10, 11) \\
C_4 = \text{XOR of bits } (4, 5, 6, 7, 12) \\
C_8 = \text{XOR of bits } (8, 9, 10, 11, 12)
\]
• Give the position of the bit in error
• Uniquely describe a bit in error \((2^k - 1 \geq n + k)\)

**Table 7.2**

<table>
<thead>
<tr>
<th>Number of Check Bits, (k)</th>
<th>Range of Data Bits, (n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2–4</td>
</tr>
<tr>
<td>4</td>
<td>5–11</td>
</tr>
<tr>
<td>5</td>
<td>12–26</td>
</tr>
<tr>
<td>6</td>
<td>27–57</td>
</tr>
<tr>
<td>7</td>
<td>58–120</td>
</tr>
</tbody>
</table>
Single-Error Correction, Double-Error Detection

- Hamming code: detect and correct only a single error
- By adding another parity bit to the coded word: correct a single error and detect double errors
- Ex 001110010100(P13)

If $C = 0$ and $P = 0$, no error occurred.
If $C \neq 0$ and $P = 1$, a single error occurred that can be corrected.
If $C \neq 0$ and $P = 0$, a double error occurred that is detected, but that cannot be corrected.
If $C = 0$ and $P = 1$, an error occurred in the $P_{13}$ bit.
7.5 Read-Only Memory (ROM)

- Stored permanent binary information

- Programmable Intersection (switch) = crosspoint

- X to denote a temporary connection

- Memory unit

- Combination circuit

**FIGURE 7.9** ROM block diagram

**FIGURE 7.10** Internal logic of a $32 \times 8$ ROM
### Table 7.3

**ROM Truth Table (Partial)**

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I_4$</td>
<td>$A_7$</td>
</tr>
<tr>
<td>$I_3$</td>
<td>$A_6$</td>
</tr>
<tr>
<td>$I_2$</td>
<td>$A_5$</td>
</tr>
<tr>
<td>$I_1$</td>
<td>$A_4$</td>
</tr>
<tr>
<td>$I_0$</td>
<td>$A_3$</td>
</tr>
<tr>
<td></td>
<td>$A_2$</td>
</tr>
<tr>
<td></td>
<td>$A_1$</td>
</tr>
<tr>
<td></td>
<td>$A_0$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$I_4$</th>
<th>$I_3$</th>
<th>$I_2$</th>
<th>$I_1$</th>
<th>$I_0$</th>
<th>$A_7$</th>
<th>$A_6$</th>
<th>$A_5$</th>
<th>$A_4$</th>
<th>$A_3$</th>
<th>$A_2$</th>
<th>$A_1$</th>
<th>$A_0$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

**5 × 32 decoder**

*Image.*
Example 7.1

Table 7.4

Truth Table for Circuit of Example 7.1

<table>
<thead>
<tr>
<th>Inputs</th>
<th>Outputs</th>
<th>Decimal</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$B_5$ $B_4$ $B_3$ $B_2$ $B_1$ $B_0$</td>
<td></td>
</tr>
<tr>
<td>0 0 0</td>
<td>0 0 0 0 0 0 0 0</td>
<td>0</td>
</tr>
<tr>
<td>0 0 1</td>
<td>0 0 0 0 0 1</td>
<td>1</td>
</tr>
<tr>
<td>0 1 0</td>
<td>0 0 0 1 0 0</td>
<td>4</td>
</tr>
<tr>
<td>0 1 1</td>
<td>0 0 1 0 0 1</td>
<td>9</td>
</tr>
<tr>
<td>1 0 0</td>
<td>0 1 0 0 0 0</td>
<td>16</td>
</tr>
<tr>
<td>1 0 1</td>
<td>0 1 1 0 0 1</td>
<td>25</td>
</tr>
<tr>
<td>1 1 0</td>
<td>1 0 0 1 0 0</td>
<td>36</td>
</tr>
<tr>
<td>1 1 1</td>
<td>1 1 0 0 0 1</td>
<td>49</td>
</tr>
</tbody>
</table>

(a) Block diagram

(b) ROM truth table