ROM MEMORY AND DECODERS

INEL4207
RANDOM ACCESS MEMORY

• Random Access Memory (RAM)
  • read and write memory

  • volatile

• Static RAM (SRAM)
  • store information as long as power is applied

  • will not lose data during a read cycle

• Dynamic RAM (DRAM)
  • uses a capacitor to store data

  • must be refreshed periodically to prevent data loss

  • read cycles destroy DRAM data (must be re-written)

• SRAM takes ~ 4 x DRAM Silicon area
READ-ONLY MEMORY (ROM)

• Non-volatile
• ROM is often needed in digital systems such as:
  – Holding the instruction set for a microprocessor
  – Firmware
  – Calculator plug-in modules
  – Cartridge style video games
A 256-MBYTE MEMORY CHIP

- Memory block contains $2^{M+N}$ storage locations
- When a bit is selected,
  - sense amplifiers: used to read/write to the RAM location
- Horizontal rows: wordlines
- Vertical lines: bitlines
READ-ONLY MEMORY (ROM)

- The basic structure of the NMOS static ROM is shown in the figure.
- The existence of a NMOS means a “0” is stored at that address otherwise a “1” is stored.
- The major downfall to this particular circuit is that it dissipates a lot of power.
Figure 16.30 A simple MOS ROM organized as 8 words × 4 bits.
READ-ONLY MEMORY (ROM)

- The domino CMOS ROM is one technique used to lower the amount of power dissipation.
NAND-ARRAY STRUCTURE ROM

- Can be directly used with NAND decoder
- Active-low word bits:
  - All W's are HIGH except selected row
  - absence of FET makes bit low;
  - presence makes bit high
NMOS NOR ADDRESS DECODERS

- Output 0 is high if both $A_0$ and $A_1$ are low

Row 0 = $(A_1 + A_0)'$
Row 1 = $(A_1 + A_0')'$
Row 2 = $(A_1' + A_0)'$
Row 3 = $(A_1' + A_0')'$
**Figure 16.25** A NOR address decoder in array form. One out of eight lines (row lines) is selected using a 3-bit address.
NMOS NAND ADDRESS DECODERS

• Output 3 is low if both A0 and A1 are high

Row 0 = \((A_1' A_0')'\)
Row 1 = \((A_1' A_0)'
Row 2 = \((A_1 \cdot A_0')'
Row 3 = \((A_1 \cdot A_0)'

[Diagram of NMOS NAND address decoders]
DOMINO CMOS ADDRESS DECODERS
PASS-TRANSISTOR COLUMN DECODER

• 3-bit column data selector using pass-transistor logic
Figure 16.27 A tree column decoder. Note that the colored path shows the transistors that are conducting when $A_0 = 1$, $A_1 = 0$, and $A_2 = 1$, the address that results in connecting $B_5$ to the data line.
Figure 16.26 A column decoder realized by a combination of a NOR decoder and a pass-transistor multiplexer.
<table>
<thead>
<tr>
<th>address $A_3A_2A_1A_0$</th>
<th>$D_0$</th>
<th>transistors &quot;ON&quot; (subscripts only)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1010</td>
<td></td>
<td></td>
</tr>
<tr>
<td>0001</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Row decoder

Column decoder

ROM

data buffer
address $A_3A_2A_1A_0$ transistors "ON" (subscripts only)

<table>
<thead>
<tr>
<th>address $A_3A_2A_1A_0$</th>
<th>$D_O$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0101</td>
<td></td>
</tr>
<tr>
<td>1000</td>
<td></td>
</tr>
<tr>
<td>0010</td>
<td></td>
</tr>
<tr>
<td>1010</td>
<td></td>
</tr>
<tr>
<td>0001</td>
<td></td>
</tr>
</tbody>
</table>