Memory Circuits

Semiconductor Memory Classification

<table>
<thead>
<tr>
<th>EWM</th>
<th>NVR/WM</th>
<th>ROM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random Access</td>
<td>Non-Random Access</td>
<td>EPROM</td>
</tr>
<tr>
<td>SRAM</td>
<td>FIFO</td>
<td>E²PROM</td>
</tr>
<tr>
<td>DRAM</td>
<td>LIFO</td>
<td>FLASH</td>
</tr>
<tr>
<td>Shift Register</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAM</td>
<td></td>
<td>Mask-Programmed</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programmable (PROM)</td>
</tr>
</tbody>
</table>

Memory Architecture: Decoders

Array-Structured Memory Architecture

Problem: ASPECT RATIO or HEIGHT >> WIDTH

Amplify swing to rail-to-rail amplitude

Selects appropriate word

Storage Cell

Bit Line

Word Line

Column Decoder

Base Decoder

Source Amplitudes / Drives

Input-Output (M bits)

N words => N select signals
Too many select signals

Decoder reduces # of select signals
$K = \log_2 N$
Hierarchical Memory Architecture

- Advantages:
  1. Shorter wires within blocks
  2. Black address activates only 1 block ==> power savings

Memory Timing: Definitions

Memory Timing: Approaches

MOS NOR ROM

DRAM Timing
Multiplexed Addressing

SRAM Timing
Self-timed
MOS NOR ROM Layout

Only 1 layer (contact mask) is used to program memory array. Programming of the memory can be delayed to one of last process steps.

Threshold raising implants disable transistors.

MOS NAND ROM

All word lines high by default with exception of selected row.

No contact to VDD or GND necessary; drastically reduced cell size. Loss in performance compared to NOR ROM.
Decreasing Word Line Delay

(a) Driving the word line from both sides
(b) Using a metal bypass
(c) Use silicides

Precharged MOS NOR ROM

PMOS precharge device can be made as large as necessary, but clock driver becomes harder to design.

Floating-Gate Transistor Programming

Avalanche injection. Removing programming voltage leaves charge trapped. Programming results in higher $V_T$. 

Flash EEPROM

Control gate Floating gate
Thin tunneling oxide

$p$-substrate
$n^+$ source $n^+$ drain
Cross-sections of NVM cells

Characteristics of State-of-the-art NVM

<table>
<thead>
<tr>
<th></th>
<th>EPROM [Tomizawa91]</th>
<th>EEPROM [Terada89, Packley89]</th>
<th>Flash EEPROM [Jinbo92]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Memory size</td>
<td>16 Mbit (0.6 ( \mu )m)</td>
<td>1 Mbit (0.8 ( \mu )m)</td>
<td>16 Mbit (0.6 ( \mu )m)</td>
</tr>
<tr>
<td>Chip size</td>
<td>7.18 x 17.39 ( \text{mm}^2 )</td>
<td>11.8 x 7.7 ( \text{mm}^2 )</td>
<td>6.3 x 18.5 ( \text{mm}^2 )</td>
</tr>
<tr>
<td>Cell size</td>
<td>3.8 ( \mu )m ( ^2 )</td>
<td>30 ( \mu )m ( ^2 )</td>
<td>3.4 ( \mu )m ( ^2 )</td>
</tr>
<tr>
<td>Access time</td>
<td>62 nsec</td>
<td>120 nsec</td>
<td>58 nsec</td>
</tr>
<tr>
<td>Erasure time</td>
<td>minutes</td>
<td>N.A.</td>
<td>4 sec</td>
</tr>
<tr>
<td>Programming time/word</td>
<td>5 ( \mu )sec</td>
<td>8 ( \mu )sec/word, 4 ( \mu )sec/chip</td>
<td>5 ( \mu )sec</td>
</tr>
<tr>
<td>Erase/Write cycles [Packley89]</td>
<td>100</td>
<td>( 10^4 )</td>
<td>( 10^3 \cdot 10^4 )</td>
</tr>
</tbody>
</table>

Read-Write Memories (RAM)

- **STATIC (SRAM)**
  - Data stored as long as supply is applied
  - Large (6 transistors/cell)
  - Fast
  - Differential

- **DYNAMIC (DRAM)**
  - Periodic refresh required
  - Small (1-3 transistors/cell)
  - Slower
  - Single Ended

Basic RAM Cell

Uses only six transistors:

Read and write use the same port. There is one wordline and two bit lines. The bit lines carry the data. The cell is small since it has a small number of wires.
6-transistor CMOS SRAM Cell

SRAM Read/Write

The key issue in an 6T SRAM is how to distinguish between read and writes. There is only one wordline, so it must be high for both reads and writes. The key is to use the fact there are two bitlines.

Read:
- Both Bit and Bit must start high. A high value on the bitline does not change the value in the cell, so the cell will pull one of the lines low.

Write:
- One (Bit or Bit) is forced low, the other is high.
- This low value overpowers the pMOS in the inverter, and this will write the cell.

RAM Cell Design

For the cell to work correctly a zero on the bit line must over power the pMOS pull up, but a one on the bit line must not over power the pull down (otherwise reads would not work)

For the pull down M3 is passing a zero, so for it to overpower the pMOS it must be at least as wide (preferably 1.5x as wide). This gives a 2:3:1 current ratio between the nMOS and the pMOS.

For pull up M3 is passing a one so it is somewhat weaker. Still M3 should be 1.5 to 2x smaller than M1 to make sure a read does not disturb the value of the cell.

6T-SRAM — Layout

There are many clever SRAM layouts. This is a common one:

This layout is fairly dense, since the most of the contacts (bitline, Vdd, Gnd) are shared. Also the a clever cross-coupling method is used.
More Cell Layout

- A conservative cell:
  - It has substrate and well connects in each cell
  - It has a wordline poly contact in each cell
  - pMOS transistors are very weak (3:3)
  - nMOS pulldown is 6:2
  - All the boundaries are shared
  - 41 x 28, about 1/4 the size of latch cell
- A slightly smaller cell
  - Only nwell contact in cell
  - pMOS transistors are very weak (3:3)
  - nMOS pulldown is 6:2
  - 36 x 28
- White box is the repeat box. Cells overlap contacts

Resistance-load SRAM Cell

Static power dissipation — Want $R_L$ large
Bit lines precharged to $V_{DD}$ to address $t_p$ problem

Dual Port RAM

Split wordline so there are two wordlines, one for each pass transistor.

- Nearly the same size as SRAM, 46 x 28
- Can read two different cells in one cycle or perform one write.
  - Raise WL1 on Register 5, and WL2 on Register 7. Register 5 value will be on bit\textsubscript{b} (complemented, of course), and Register 7 will be on bit. Since you need both bit and bit\textsubscript{b} to write the cell, you can only do one write per cycle.

Multiport RAM Cell

Can build true multi-ported memory cell, by adding more bitline pairs and wordlines to a cell.

- Shown in the figure is a true dual port cell. You can read or write on each port every cycle. Since it has more bitlines than the previous cell, it is much larger in area.
Topic 10 - 29 Nov-22-10 E4.20 Digital IC Design

3-Transistor DRAM Cell

No constraints on device ratios
Reads are non-destructive
Value stored at node X when writing a “1” = \( V_{WWL} - V_{TH} \)

1-Transistor DRAM Cell

Write: \( C_{S} \) is charged or discharged by asserting WL and BL.
Read: Charge redistribution takes place between bit line and storage capacitance
\[
\Delta V = V_{BL} - V_{PRB} = \frac{C_{S} - C_{BL}}{C_{S} + C_{BL}} \times (V_{DD} - V_{TH})
\]
Voltage swing is small: typically around 250 mV.

3T-DRAM — Layout

1T DRAM requires a sense amplifier for each bit line, due to charge redistribution read-out.
DRAM memory cells are single ended in contrast to SRAM cells.
The read-out of the 1T DRAM cell is destructive; read and refresh operations are necessary for correct operation.
Unlike 3T cell, 1T cell requires presence of an extra capacitance that must be explicitly included in the design.
When writing a “1” into a DRAM cell, a threshold voltage is lost.
This charge loss can be circumvented by bootstrapping the word lines to a higher value than \( V_{DD} \).

DRAM Cell Observations
1-T DRAM Cell

(a) Cross-section
(b) Layout

Used Polysilicon-Diffusion Capacitance
Expensive in Area

Advanced 1T DRAM Cells

Address Transition Detection

SEM of poly-diffusion capacitor 1T-DRAM
Row Decoders

Collection of $2^m$ complex logic gates
Organized in regular and dense fashion

(N)AND Decoder

$$WL_0 = A_0A_1A_2A_3A_4A_5A_6A_7A_8A_9$$

$$WL_{511} = A_0A_1A_2A_3A_4A_5A_6A_7A_8A_9$$

NOR Decoder

$$WL_0 = A_0 + A_1 + A_2 + A_3 + A_4 + A_5 + A_6 + A_7 + A_8 + A_9$$

$$WL_{511} = A_0 + A_1 + A_2 + A_3 + A_4 + A_5 + A_6 + A_7 + A_8 + A_9$$

Dynamic Decoders

Dynamic 2-to-4 NOR decoder 2-to-4 MOS dynamic NAND Decoder

Propagation delay is primary concern

A NAND decoder using 2-input pre-decoders

4 input pass-transistor based column decoder

Advantage: speed ($t_{pd}$ does not add to overall memory access time)
only 1 extra transistor in signal path
Disadvantage: large transistor count
### Bitline I/O Circuit - Read

For reads both bitlines must be high, for write you need to drive the bitlines to the correct value.

- Bitlines need to be precharged, or use a pseudo nMOS load
- We will use a precharged structure:
  - To avoid a conflict during precharge, make wordline a qualified clock.

### Bitline I/O Circuit - Write

Similar to read, but need to drive the bit lines too.

- Data\_s1
  - Instead of \( \bar{D} \) could be AND\_ed with \( \bar{Q} \)

Notice that since the memory cell is a storage element, its enable (the wordline) needs to be a \( _Q \) signal. That will ensure that the clock falls latching in the data BEFORE the data has a chance to change. The wordline is really the clock to the latch (memory) cell.

Need to isolate the write driver so it does not fight with the precharge (power issue), which is why the write signal is qualified.
Write Drivers

Also need to worry about the series resistance of the driver

- The resistance of the 3 series nMOS transistors must still be 2x less than the resistance of the pMOS in the cell

- If the pass device in the cell is 4:2, and the pMOS load is 3:4, then each nMOS in the driver must be at least 8:2. If the pMOS was 3:2, it would be hard to get the cell to write in isim.

Bitline Multiplexing

The bitline pitch is pretty small (about 28λ) and there is a lot of stuff that is needed for the bitline (read and write circuits). Often many bitlines are muxed together, and one set of IO circuits is used for these bitlines

Two basic options:
- Share mux between read and write circuits
  - Least amount of logic needed on bit pitch
    - Adds another series device to write drivers, need to use single write device
- Use different mux for read and write
  - No series devices in write path
    - Write mux can be qualified with Write & Clock
    - Adds some more muxes to bit logic

Bitline Mux - Option 1

Should have a precharge on the output of the mux too, since otherwise the output will have a degraded high level.
**Bitline Mux - Option 2**

Uses separate mux for read and write

Notice that the read mux is precharged

- You don’t need to use pMOS devices in the mux

```
Data_s1
A0_s1
WriteOdd_q1
```

**Sense Amplifiers**

- Since Cdiff ~ Cgate, the diffusion contacts of access devices cause large cap on the bitlines, for large arrays
- Bitline cap becomes an issue around 32 cells/bitline
  
  Example:
  If the diffusion contacts are shared (adjacent cells), 128 cells @4fF/
  2cells = 256fF + wire cap. This would lead to access times of around
  3ns.

  Can take advantage of the differential nature of the bitlines
  
  - Decrease delay by sensing smaller signals
  - Noise margin is ok, most noise is common mode
  
  Build a differential amplifier (sense amp)
  
  (Not needed in this class)

**Sense Amp Circuit**

**Latch-Based Sense Amplifier**

Initialized in its meta-stable point with EQ

Once adequate voltage gap created, sense amp enabled with SE

Positive feedback quickly forces output to a stable operating point.
**Single-to-Differential Conversion**

How to make good $V_{\text{ref}}$?

**Open bitline architecture**

**DRAM Read Process with Dummy Cell**

(a) reading a zero
(b) reading a one
(c) control signals

**Single-Ended Cascode Amplifier**

$V_{\text{DD}}$

$V_{\text{ref}}$

$WLC$

$WL$
Programmable Logic Array

Pseudo-Static PLA

Dynamic PLA

Clock Signal Generation for self-timed dynamic PLA

(a) Clock signals
(b) Timing generation circuitry.
Programmable Logic Array
structured approach to random logic
“two-level logic implementation”
NOR–NOR (product of sums)
NAND–NAND (sum of products)

IDENTICAL TO ROM!

Main difference
-ROM: fully populated
-PLA: one element per minterm

Note: Importance of PLA’s has drastically reduced
1. slow
2. better software techniques (multi-level logic synthesis)