### Memory

<table>
<thead>
<tr>
<th>Access</th>
<th>Non-Access</th>
<th>EPROM</th>
<th>E'PROM</th>
<th>Mask-Programmed</th>
<th>Programmable (PROM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>Non-Random</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SRAM</td>
<td>FIFO</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DRAM</td>
<td>LIFO</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Shift Register</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>CAM</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Memory Decoders

- **Problem:** Aspect Ratio or Height >> Width
- **Solution:**
  - Amplify swing to rail-to-rail amplitude
  - Selects appropriate word

### Array-Structured Memory

- **Problem:** Aspect Ratio or Height >> Width

### Array Decoding

- Typically want an aspect ratio that is not too far from square
- How to divide up the row, column address decoding?

  Use an 8K x 32 SRAM = 256 Kb = 2^15

  2^15 = 2^k rows x 2^k columns

  Row decoder is 9 to 512 decoder

  Every 32 (2^k) columns is a ‘word’, and we only need to decode words. So, column decoder needs to decode 2^k words, so need a 4 to 16 column decoder.

### Hierarchical Memory Arrays

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

### Memory Timing Definitions

- **Read Cycle**
- **Write Cycle**
Memory Timing Approaches

- **Address Bus**
- **Row Address**
- **Column Address**

**RAS**

**CAS**

**RAS-CAS Timing**

- **DRAM Timing**
- **SRAM Timing**
- **Multiplexed Addressing**
- **Self-timed**

---

**HM6264 Interface**

- **Multiplexed Addressing**
- **Self-timed**

---

**Example: HM6264 8kx8 SRAM**

- **Address**
- **Bus Address**
- **Address transition**

---

**Function Table**

- **Pin Description**
- **Function Table**

---

**Timing**

- **Read Cycle 1**

---
What Does All This Mean

- For a read:
  - If you assert CS1, CS2, address, and OE all at the same time, it will be max 85ns before valid data are available at chip outputs.
- For a write:
  - You can assert CS1, CS2, address, data, and WE all at the same time if you want to.
  - You need to wait 55ns from WE edge, or 75ns from CS1/CS2 edge for write to have happened.

R/W Memories In General

- **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

SRAM Circuits

- 6-transistor SRAM Cell

SRAM Cell, Transistors

- 6-transistor SRAM Cell

SRAM, Resistive Pullups

- 6-transistor SRAM Cell

Array-Structured Memory

- Problem: ASPECT RATIO or HEIGHT >> WIDTH

- Amplify swing to rail-to-rail amplitude
- Selects appropriate word
Memory Column

- Each column has all the support circuits.

Reading the Bit

- Single-ended read using an inverter.
- Dynamic pre-charge on the bit lines.
- P-types pull bit lines high.

Reading the Bit 2

- Single-ended read using an inverter.
- Dynamic pre-charge on the bit lines.
- Note the N-types used as pull-ups.

Reading the Bit 3

- Differential read using sense amp.
- Static N-type pullup on the bit lines.

Read Waveforms

Sense Amp

- Current-mirror SA

Only one bit line will swing.

Want Sense Amplifier turned on for short amount of time in order to save power.

Job of SA is to sense bit line swing, amplify to full swing output.
Sense Amp Transistors

Column Organization

Write Circuits

Write Circuit Simulation

Analog Sim, Circuit

Analog Analysis, Write

Long-channel FET used as current "source"

\[
\begin{align*}
V_{DD} & \approx V_{DD2} - \frac{k_n M_6}{W/L} \left( VT_n - \frac{V_{DD}}{2} \right) \\
V_{DD} & \approx V_{DD2} - \frac{k_p M_4}{W/L} \left( VT_p - \frac{V_{DD}}{2} \right) \\
V_{DD} & \approx V_{DD2} - \frac{2k_n M_5}{W/L} \left( VT_n - \frac{V_{DD}}{2} \right) \\
V_{DD} & \approx V_{DD2} - \frac{2k_p M_1}{W/L} \left( VT_p - \frac{V_{DD}}{2} \right)
\end{align*}
\]
Analog Analysis, Read

\[ \frac{k_{n,M}}{2} \left( \frac{V_{DD} - \left( V_{DD} \frac{V_{DD}}{2} \right)}{2} \right) = \frac{k_{n,M}}{2} \left( V_{DD} - \frac{V_{DD}}{2} \right) \]

\[ \frac{W}{L} \text{, } V_{TH} < 10 \left( \frac{W}{L} \right) \]

Another 6T SRAM Layout

6T SRAM Layout

Another 6T SRAM Layout

Array-Structured Memory

Problem: ASPECT RATIO or HEIGHT >> WIDTH

SRAM bit from makemem (v1)

SRAM bit from makemem (v2)
Row Decoders

- Select exactly one of the memory rows
- Simple versions are just gates

Row Decoder Gates

- Standard gates
- Or, pseudo-nmos gates with static pull up
- Easier to make large fan-in NOR

Pre-decode Row Decoder

- Multiple levels of decoding can be more efficient layout

Pre-decode Row Decoder

- Other circuit tricks for building row decoders…

Array-Structured Memory

Problem: ASPECT RATIO or HEIGHT >> WIDTH

- Amplify swing to rail-to-rail amplitude
- Selects appropriate word
Sharing Sense Amps

Sharing Sense Amplifiers

Limited swing bit lines

Col Addr

Pass Transistor Column Decode (Tree decode)

SA

SA

SA

Additional Column Decode (gate-based)

Col Addr

Data out

Full swing signals

Sense Amp Mux

4 to 1 Tree Decoder (pg. 595, Rabaey)

A0

A0'

A1

A1'

BL0

BL1

BL0

BL1

Need to use pass transistors because of limited swing.

Number of pass transistors in series is a concern, but limited swing helps speed.

Sense Amp Mux

Decoded Column Decode

Very common to require multiple read ports

Think about a register file, for example

Improving Speed, Power

Critical path runs through row decode, word line assertion

- Need smaller decoding, less word line capacitance in order to improve speed.
- Break a large array into smaller sub-arrays, and use hierarchical decoding to select a sub array
  - PowerPC 32K x 8 cache broken into 32 blocks, each 1K x 8
  - Cypress 1Mbit Dual Port broken into 32 blocks, each 32 K bits
    (2^2 x 2^11 = 2^14). Each block is 512 rows x 64 columns
  - Mitsubishi SRAM (Rabaey text). 32 blocks of 128K bits (1024 rows x 128 columns)
- Only one sub-array will be activated, saves power!!!!

Multi-Port Memory
- Slightly larger cell, but with single-ended read – makes a great register file

- Get rid of the pull-ups!
  - Store info on capacitors
  - Means that stored information leaks away

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

- Once you agree to use a capacitor for charge storage there are other ways to build this…
1T DRAM Circuit

Row select line

Vdd

2 Transistor DRAM Cell

Storage Capacitor

Vdd

Column bit line

C_{storage} = C_{drain} + C_{gate} + C_{source}

1T DRAM Observations

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}.

1T DRAM Read/Write

Write: C_s is charged or discharged by asserting WL and BL.

Read: Charge redistribution takes place between bit line and storage capacitance

\[ \Delta V = \frac{V_{BL} - V_{PRE} - (V_{ATT} - V_{PRE})}{C_s + C_{tot}} \]

Voltage swing is small; typically around 250 mV.

1T DRAM Cell

Array of DRAM Cells
Reading a 1T DRAM Cell

- **Charge Sharing**

\[ C_{eq} = 75 \text{fF} \]
\[ C_{eq} = 500 \text{fF} \]

- **DRAM Sense Amp**

  - Column bit lines are precharged to Vdd/2.
  - Near proximity of column lines gives excellent common noise rejection from coupled signals.
  - Sense amp must discriminate less than 100 nV voltage difference in the column lines.

Photo of 1T DRAM

Examples of Advanced DRAMs

- **Cell Plate Si**
- **Capacitor Insulator**
- **Storage Node Poly**
- **2nd Field Oxide**
- **Refilling Poly**
- **Si Substrate**
- **Trench Cell**
- **Stacked-capacitor Cell**

Advanced DRAM Cells

- **Trench Capacitor**

Advanced DRAM Cells

- **ONO – Oxide-Nitride-Oxide**
- Special IC process for making DRAM
- Smaller than 2 transistor cell... higher density memory

Memory Timing Approaches

- **MSB**
- **LSB**

- **Address Bus**
  - Row Address
  - Column Address

- **RAS**
  - Address transition initiates memory operation

- **CAS**

**DRAM Timing**
- Multiplexed Addressing

**SRAM Timing**
- Self-timed
**DRAM Interface**

Multiplexed Address bits (Row, Column), RAS# (Row Address Strobe), CAS# (Column Address Strobe) used to latch in address

**Comments on Timing**

- Typical times are Tras = 60 ns (RAS pulse width), Tce = 100 ns
  - Extra time on Read cycle (RAS high) is needed to recharge bit lines
- Block mode transfers (Page mode transfers) read bits from same row
  - Only change column address
  - Time to first bit on row = 50 ns, time to successive bits = 25 ns (we have access to all bits on this row, just need to mux them out).

**Architectural Issues**

- Need to support block transfers efficiently since DRAM used as main memory and reads/writes due to cache fills
- Add a clock to DRAM interface (SDRAM, DDR-SDRAM) to support burst mode operations for cache fills
  - Pentium burst mode is 2-1-1-1-1 (two clocks for first data, 1 clock for each successive data, address only provided for first data, internal counter on RAM used for address generation).
  - Pentium Pipelined burst mode is:
    2-1-1-1; 1*1-1-1; 1*1-1-1; ... Successive cycles pick up where the last cycle left off.

**SDRAM - Use CAS for Bursts**

**DDR SDRAM**

- Differential Clocks
- Double Data Rate
**DRAM Timing**

- Clock Frequency – 133 Mhz, 100 Mhz
- Two clock latency to first data (20 ns for 100 Mhz clock)
  - SDRAM - 10 ns per location afterwards. For byte-wide, 100 MB/sec transfer rate. 400 MB/sec on 32-bit bus
  - DDR-SDRAM - 5 ns per location afterwards. For byte-wide, 200 MB/sec transfer rate. On 32-bit bus, 800 MB/sec transfer rate.

**RAMBUS DRAM (RDRAM)**

- DRAM with a high speed interface
- 400 Mhz differential clock, data transferred on each edge
- Reduced swing signaling about a reference voltage
  - Termination voltage is 1.5 V
  - Reference Voltage is 1.0 V
  - Signals swing +/- 200 mv about reference voltage
- All traces are transmission lines

**RDRAM Bandwidth**

- External bus is 18 bits wide (2 bytes + 2 parity bits)
- External clock cycle is 400 Mhz, but data is clocked on each edge
  - Actually, external clock is a differential pair and data is sampled at each crossing
- Total Bandwidth is 1.6 GBytes/s
  - 2 bytes * 400 Mhz * 2 edges => 1.6 Gbytes
  - Initial configurations are 4 M x 18 (72 Mbits)

**Maximum Bandwidth**

- Note that maximum bandwidth with one RDRAM controller is 1.6GB/s.
  - Only one RDRAM chip can be active at a time on RDRAM bus.
  - More RDRAM chips increase capacity, not bandwidth.
  - With normal DRAM and SDRAM, can increase bandwidth by just adding more DRAM chips in parallel from same DRAM controller
  - To double the bandwidth, would need two separate RDRAM controllers

**Normal Bus for DRAM DIMMs**

[Diagram of DRAM DIMMs]

**RDRAM Bus**

[Diagram of RDRAM bus]

Signaling Technology for RDRAM basically the same as Pentium III bus. RDIMMs must be connected serially to avoid stubs.
Deep Pipelining - High Latency

16 bytes transferred because 4 clocks * 2 edges * 2 bytes/transfer (external bus is 16 or 18 bits wide). 20 clock latency, 20 ns from column address.

RDRAM Addressing

- 3-Bit Row bus used to give commands to RDRAM
- ROW Activate command used for read
  - 4 clocks transfers 8 groups of 2 bits over Row bus due to dual edge clocking (24 bits total)
  - 24 bits in Row Activate command split between device address (6 bits), bank select (4 bits), row select (9 bits), and reserved bits
- There are no chip select lines, internal register holds device address
  - All chips monitor bus - if bus device address matches internal id, then chip is selected.

Row Activate Command

RDRAM System Arch

18 bit wide external data bus which expands into 128 bit wide datapath internal to chip.

RDRAM Internal Arch

Regular DRAM

- Multiple Banks are key to high throughput
- As one DRAM bank is recovering from read operation, next bank is being accessed
- Essentially on-chip memory interleaving
- Goal is to hide latency and bitline precharge time (recovery time)
  - Latency is access to first byte, critical path through row-decode and word line assertion
  - Bitline Precharge time (recovery time to next access) depends on number of bits in a column (number of rows)
Single Bank DRAM

Multi-Bank DRAM

Peak Bandwidth

ROM

Fig. 11. Multibank system bandwidth calculations.

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

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

- EPROM
  - Erasable Programmable ROM
- EEPROM
  - Electrically Erasable Programmable ROM
- Flash EEPROM
  - Electrically Erasable Programmable ROM that is erased in large chunks
- All these devices rely on trapping charge on a floating gate

EPROM

- (a) Device cross-section
- (b) Schematic symbol

Programming EPROM

- Avalanche injection.
- Removing programming voltage leaves charge trapped.
- Programming results in higher $V_{th}$.
- Higher $V_{th}$ (around 7V) means that 5V $V_{gs}$ no longer turns on the transistor
- $SiO_2$ is an excellent insulator
- Trapped charge can stay for years

Erasing an EPROM

- Erase by shining UV light through window in the package
- UV radiation makes oxide slightly conductive
- Erasure is slow - from seconds to minutes depending on UV intensity
- Also the erase/program cycles are limited (around 1000), mainly as a result of the UV erasing
- But, EPROMs are simple and dense

EEPROM

- Two transistors instead of one
  - The second keeps you from removing too much charge during erasure
- Bigger and not as dense as EPROM
- But, more erase/program cycles
  - On the order of $10^5$
  - Eventually you get permanently trapped charge in the $SiO_2$

EEPROM

- (a) Flotox transistor
- (b) Fowler-Nordheim I-V characteristic
- (c) EEPROM cell during a read operation
- Thin oxide allows erasing in-system
  - Fowler-Nordheim Tunneling
Flash EEPROM

- Essentially the same as EEPROM
- But, large regions erased at once
- Means you can monitor the voltages and don’t need the extra access transistor

Realistic PROM Devices

<table>
<thead>
<tr>
<th></th>
<th>EPROM</th>
<th>EPROM</th>
<th>Flash EPROM</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>[Tomlin91]</td>
<td>[Terzic90, Pad Ley92]</td>
<td>[Jiao92]</td>
</tr>
<tr>
<td>Memory size</td>
<td>16 Mbit (0.6 μm)</td>
<td>1 Mbit (0.8 μm)</td>
<td>16 Mbit (0.6 μm)</td>
</tr>
<tr>
<td>Chip size</td>
<td>7.18 x 7.39 mm²</td>
<td>11.5 x 7.7 mm²</td>
<td>0.5 x 18.5 mm²</td>
</tr>
<tr>
<td>Cell size</td>
<td>3.8 μm²</td>
<td>39 μm²</td>
<td>3.4 μm²</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 μsec</td>
<td>8 μsec/word, 4 sec/chip</td>
<td>5 μsec</td>
</tr>
<tr>
<td>Erase/Write cycles</td>
<td>100</td>
<td>$10^7$</td>
<td>$10^8$-$10^9$</td>
</tr>
</tbody>
</table>

Content Addressable Mem

- Asks the question: Are there any locations that hold this value?
- Used for tag memories in associative caches
- Or translation lookaside buffers
- Or other pattern matching applications

Content Addressable Mem

- Add the Match line
- Essentially a distributed NOR gate
Programmable Logic Array

> Still useful for random combinational logic
> Standard cell ASIC tools may be replacing them
> They can generate dense AND-OR circuits

Pseudo-Static PLA Circuit

Dynamic PLA

PLA Layout

PLA vs. ROM

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)
FPGAs

- Field Programmable Gate Arrays
  - Array of P-type and N-type transistors
  - Sources and drains connected to
    - Power and ground
    - Metal
  - Map gate structures to sea of gates
  - Less expensive – only modify metal masks