### Memory

<table>
<thead>
<tr>
<th>RWM</th>
<th>NVRWM</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></td>
<td>Shift Register</td>
<td></td>
</tr>
<tr>
<td></td>
<td>CAM</td>
<td></td>
</tr>
</tbody>
</table>

### Memory Decoders

- **N Words**
  - **M bits**
  - Storage Cell
  - Input-Output (M bits)

- **Word 0**
- **Word 1**
- **Word 2**
- **Word N-2**
- **Word N-1**

- **N words => N select signals**
- **Too many select signals**

- **Decoder reduces # of select signals**
  - $K = \log_2 N$
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 = $2^{18}$

  $2^{18} = 2^9$ rows x $2^9$ columns
  
  Row decoder is 9 to 512 decoder

  Every 32 ($2^5$) columns is a ‘word’, and we only need to decode words. So, column decoder needs to decode $2^4$ 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
READ
Read Access
Data Valid
Write Access
DATA
Data Written
WRITE
Read Access
Write Cycle
Memory Timing Approaches

<table>
<thead>
<tr>
<th>Address Bus</th>
<th>Row Address</th>
<th>Column Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>RAS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CAS</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

RAS-CAS timing

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

Example: HM6264 8kx8 SRAM

![Diagram of HM6264 8kx8 SRAM circuit](image)
HM6264 Interface

Pin Description

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Function</th>
<th>Pin name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>A0 to A12</td>
<td>Address input</td>
<td>WE</td>
<td>Write enable</td>
</tr>
<tr>
<td>I/O1 to I/O8</td>
<td>Data input/output</td>
<td>OE</td>
<td>Output enable</td>
</tr>
<tr>
<td>CS1</td>
<td>Chip select 1</td>
<td>NC</td>
<td>No connection</td>
</tr>
<tr>
<td>CS2</td>
<td>Chip select 2</td>
<td>VCC</td>
<td>Power supply</td>
</tr>
</tbody>
</table>

Function Table

<table>
<thead>
<tr>
<th>WE</th>
<th>CS1</th>
<th>CS2</th>
<th>OE</th>
<th>Mode</th>
<th>VCC current</th>
<th>I/O pin</th>
<th>Ref. cycle</th>
</tr>
</thead>
<tbody>
<tr>
<td>×</td>
<td>H</td>
<td>×</td>
<td>×</td>
<td>Not selected (power down)</td>
<td>I_{cc}, I_{ls1}</td>
<td>High-Z</td>
<td>—</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>L</td>
<td>×</td>
<td>Not selected (power down)</td>
<td>I_{cc}, I_{ls1}</td>
<td>High-Z</td>
<td>—</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>Output disable</td>
<td>I_{cc}</td>
<td>High-Z</td>
<td>—</td>
</tr>
<tr>
<td>H</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>Read</td>
<td>I_{cc}</td>
<td>Dout</td>
<td>Read cycle (1)–(3)</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>Write</td>
<td>I_{cc}</td>
<td>Din</td>
<td>Write cycle (1)</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>H</td>
<td>L</td>
<td>Write</td>
<td>I_{cc}</td>
<td>Din</td>
<td>Write cycle (2)</td>
</tr>
</tbody>
</table>

Note: ×: H or L
## Timing

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Read cycle time</td>
<td>$t_{RC}$</td>
<td>85</td>
</tr>
<tr>
<td>Address access time</td>
<td>$t_{AA}$</td>
<td>85</td>
</tr>
<tr>
<td>Chip select access time</td>
<td>CST $t_{CO1}$</td>
<td>85</td>
</tr>
<tr>
<td></td>
<td>CSZ $t_{CO2}$</td>
<td>85</td>
</tr>
<tr>
<td>Output enable to output valid</td>
<td>$t_{OE}$</td>
<td>45</td>
</tr>
<tr>
<td>Chip selection to output in low-Z</td>
<td>CST $t_{LZ1}$</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>CSZ $t_{LZ2}$</td>
<td>10</td>
</tr>
<tr>
<td>Output enable to output in low-Z</td>
<td>$t_{ULZ}$</td>
<td>5</td>
</tr>
<tr>
<td>Chip de-selection in to output in high-Z</td>
<td>CST $t_{HZ1}$</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>CSZ $t_{HZ2}$</td>
<td>0</td>
</tr>
<tr>
<td>Output disable to output in high-Z</td>
<td>$t_{OHz}$</td>
<td>0</td>
</tr>
<tr>
<td>Output hold from address change</td>
<td>$t_{OH}$</td>
<td>10</td>
</tr>
</tbody>
</table>

### Read Cycle 1

- Address
- Valid address
- $t_{AA}$
- CST $t_{CO1}$
- CSZ $t_{CO2}$
- $t_{LZ1}$
- $t_{LZ2}$
- $t_{OE}$
- $t_{ULZ}$
- CST $t_{HZ1}$
- CSZ $t_{HZ2}$
- $t_{OHz}$
- $t_{OH}$

$(\overline{WE} = \overline{V_{IH}})$
Read Cycle 1

\( t_{RC} \leq 85\text{ns} \) min

\( t_{AA} \leq 85\text{ns} \) max

\( t_{OE} \leq 85\text{ns} \) max

\( t_{Z2} \geq 10\text{ns} \) min

\( t_{OLZ} \geq 5\text{ns} \) min

\( t_{CHZ} \geq 30\text{ns} \) min

\( t_{OH} \geq 10\text{ns} \) min

\( \overline{WE} = V_{IH} \)

Read Cycle 2

\( t_{AA} \)

\( t_{OH} \)

\( t_{OE} \)

\( t_{OLZ} \)

\( t_{CHZ} \)

\( t_{OH} \)

\( \overline{WE} = V_{IH}, \overline{OE} = V_{IL} \)
Read Cycle 2

<table>
<thead>
<tr>
<th>Address</th>
<th>Valid address</th>
<th>t_{AA} 85\text{ns max}</th>
<th>t_{CH} 10\text{ns min}</th>
<th>t_{CE} 10\text{ns min}</th>
<th>Dout</th>
<th>Valid data</th>
</tr>
</thead>
</table>

\[ (WE = V_{IH}, OE = V_{IL}) \]

Write Timing

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Symbol</th>
<th>HM6264B-8L</th>
<th>HM6264B-10L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Write cycle time</td>
<td>t_{WC}</td>
<td>85 — 100 ns</td>
<td>— 100 ns</td>
</tr>
<tr>
<td>Chip selection to end of write</td>
<td>t_{WE}</td>
<td>75 — 80 ns</td>
<td>— 80 ns</td>
</tr>
<tr>
<td>Address setup time</td>
<td>t_{AD}</td>
<td>0 — 0 ns</td>
<td>— 0 ns</td>
</tr>
<tr>
<td>Address valid to end of write</td>
<td>t_{AV}</td>
<td>75 — 80 ns</td>
<td>— 80 ns</td>
</tr>
<tr>
<td>Write pulse width</td>
<td>t_{WP}</td>
<td>55 — 60 ns</td>
<td>— 60 ns</td>
</tr>
<tr>
<td>Write recovery time</td>
<td>t_{WR}</td>
<td>0 — 0 ns</td>
<td>— 0 ns</td>
</tr>
<tr>
<td>WE to output in high-Z</td>
<td>t_{WZ}</td>
<td>0 — 30 ns</td>
<td>0 — 35 ns</td>
</tr>
<tr>
<td>Data to write time overlap</td>
<td>t_{DZ}</td>
<td>40 — 40 ns</td>
<td>— ns</td>
</tr>
<tr>
<td>Data hold from write time</td>
<td>t_{DH}</td>
<td>0 — 0 ns</td>
<td>— ns</td>
</tr>
<tr>
<td>Output active from end of write</td>
<td>t_{OW}</td>
<td>5 — 5 ns</td>
<td>— ns</td>
</tr>
<tr>
<td>Output disable to output in high-Z</td>
<td>t_{OD}</td>
<td>0 — 30 ns</td>
<td>0 — 35 ns</td>
</tr>
</tbody>
</table>

Notes:
1. A write occurs during the overlap of a low CS1, and high CS2, and a high WE. A write begins at the latest transition among CST going low, CS2 going high and WE going low. A write ends at the earliest transition among CST going high CS2 going low and WE going high. Time t_{WP} is measured from the beginning of write to the end of write.
Write Cycle

Address

Valid address

0E
tCW
tWR

CST

CS2
tAS
tCW

tWP

WE

tCHZ

Dout

High Impedance

Din

High Impedance

Valid data

Note: 1. If CST goes low or CS2 goes high simultaneously with WE going low or after WE going low, the outputs remain in the high impedance state.

Write Cycle

Address

Valid address

0E

75ns min 0ns min

tCW
tWR

CST

CS2
tAS 75ns min
tCW 75ns min

WE

0ns min 55ns min

0ns min, 30ns max

Dout

High Impedance

40ns min

Din

High Impedance

Valid data

Note: 1. If CST goes low or CS2 goes high simultaneously with WE going low or after WE going low, the outputs remain in the high impedance state.
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 Analysis, Write**

\[
k_{n,M6}(V_{DD} - V_{Tn}) = k_{n,M4}(V_{DD} - V_{Tn} + \frac{V_{DD}}{2} - \frac{V_{DD}}{8})
\]

\[
k_{n,M5}(V_{DD} - V_{Tn}) = k_{n,M1}(V_{DD} - V_{Tn} + \frac{V_{DD}}{2} - \frac{V_{DD}}{8})
\]

\[
(W/L)_{n,M6} \geq 0.33 (W/L)_{p,M4}
\]

\[
(W/L)_{n,M5} \geq 10 (W/L)_{n,M1}
\]
Analog Analysis, Read

\[ \frac{k_{n,M5}/V_{DD} - V_{Tn}(V_{DD}/2)^2}{(W/L)_{n,M5}} = k_n M_1 \left( (V_{DD} - V_{Tn}) \frac{V_{DD}}{2} - \frac{V_{DD}^2}{8} \right) \]

\( (W/L)_{n,M5} \leq 10 (W/L)_{n,M1} \)

6T SRAM Layout
Another 6T SRAM Layout

SRAM bit from makemem (v1)
Array-Structured Memory

Problem: ASPECT RATIO or HEIGHT >> WIDTH
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

Input-Output (M bits)

Array-Structured Memory
Sharing Sense Amps

Sharing Sense Amplifiers

Col Addr

Pass Transistor Column Decode (Tree decode)

Limited swing bit lines

SA

SA

SA

Full swing signals

Additional Column Decode (gate-based)

Col Addr

Data out

SA shared among multiple columns

Sense Amp Mux

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

A0’

BL0

BL1

A0

BL0

BL1

A1’

Need to use pass transistors because of limited swing.

A1

Number of pass transistors in series is a concern, but limited swing helps speed.
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 1Mb Dual Port broken into 32 blocks, each 32 K bits
    \((2^5 \times 2^5 \times 2^{10} = 2^{20})\). Each blocks 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

- Very common to require multiple read ports
  - Think about a register file, for example
Slightly larger cell, but with single-ended read – makes a great register file
Dynamic RAM

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

Dynamic RAM...

- Once you agree to use a capacitor for charge storage there are other ways to build this...
No constraints on device ratios
Reads are non-destructive
Value stored at node X when writing a “1” = \( V_{WWL} - V_{Tn} \)
1 T DRAM Circuit

Row select line
Vdd

Row select line

Storage Capacitor
Vdd

Column bit line

2 Transistor DRAM Cell

Equivalent Circuit

\[ C_{storage} = C_{drain} + C_{gate} + C_{source} \]

2-T (1-T) DRAM layout

› Note the increased gate size of the storage transistor
  › Increases the capacitance
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 places between bit line and storage capacitance.

$$\Delta V = V_{BL} - V_{PRE} = (V_{BIT} - V_{PRE}) \frac{C_S}{C_S + C_{BL}}$$

Voltage swing is small; typically around 250 mV.
1T DRAM Cell

Array of DRAM Cells

“Folded bit line”
Reading a 1T DRAM Cell

Charge Sharing

Row select line

$C_{\text{storage}} = 75 \text{ fF}$

Vdd

$C_{\text{bit}} = 500 \text{ fF}$

Column bit line

Precharge

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 mV voltage difference in the column lines.
Try to get more capacitance per unit area...

ONO = Oxide Nitrate Oxide

Special IC process for making DRAM

Smaller than 2 transistor cell... higher density memory
Examples of Advanced DRAMs

- Cell Plate Si
- Capacitor Insulator
- Storage Node Poly
- 2nd Field Oxide
- Si Substrate
- Word line
- Insulating Layer
- Cell plate
- Capacitor dielectric layer
- Transfer gate
- Isolation
- Storage electrode

Trench Cell

Stacked-capacitor Cell

Memory Timing Approaches

- Address Bus
  - Row Address
  - Column Address

- RAS
- CAS

DRAM Timing
- Multiplexed Addressing

SRAM Timing
- Self-timed
DRAM Interface

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

READ CYCLE

Extended Data Out Page Mode

Block transfer. Access different bits on same row, change column address.
Comments on Timing

- Typical times are Tras = 60 ns (RAS pulse width), Trc = 100 ns
  - Extra time on Read cycle (RAS high) is needed to recharge bitlines
- 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 (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

Burst Read Operation (Burst Length = 4, CAS latency = 1, 2, 3)

- **CLK**: Timing signal for the operation.
- **COMMAND**: Read A, NOP, NOP, NOP, NOP, NOP, NOP, NOP.
- **CAS latency = 1**: DOUT A0, DOUT A2, DOUT A3.
- **CAS latency = 2**: DOUT A0, DOUT A2, DOUT A3, DOUT A3.
- **CAS latency = 3**: DOUT A0, DOUT A2, DOUT A3, DOUT A3.

---

**DDR SDRAM**

- **Differential Clocks**
- **Two clock latency**
- **Data transferred on each clock crossing**
- **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

RDRAM Bus

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

Figure 7. Direct RDRAM and leaved memory transactions at full-memory bandwidth (16 bytes/10 ns).

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 3 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

- CTM/CFM
- ROW2: DR4T, DR2, BR0, BR3, RsvR
- ROW1: DR4F, DR1, BR1, RsvB
- ROW0: DR3, DR0, BR2, RsvB

10 ns

R bits = row select
DR bits = device address
BR bits = bank select
ROWA Packet

RDRAM System Arch

- Controller
- RDRAM 1
- RDRAM 2
- RDRAM n

Bus data [18:0]
RC[7:0]
RC[1:0]
TC[1:0]
VREF
Gnd(32/18)
VDC(4)

400 MHz

Figure 3. Direct RDRAM system.

18 bit wide external data bus which expands into 128 bit wide datapath internal to chip
RDRAM Internal Arch

Portion of internal architecture (4M x 16 or 4M x 18)

16 banks of 512 rows of 64 dualocts (1 dualoct = 16 bytes = 128 bits)

\[2^4 \text{(banks)} \times 2^9 \text{(rows)} \times 2^6 \text{(dualocts)} \times 2^7 \text{(one dualoct)} = 2^{26} \text{(64 Mbit)}\]

A dualoct is the smallest addressable unit.

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**

```
+----------------+-----------------+-------------------+
| CLK            | row access      | precharge         |
|                | col             |                   |
+----------------+-----------------+-------------------+
```

**Multi-Bank DRAM**

```
+----------------+-----------------+-------------------+
| Bank #1        | row access      | precharge         |
|                | col             |                   |
| Bank #2        | row access      |                   |
|                | col             |                   |
+----------------+-----------------+-------------------+
```

Number of banks required to hire all row latency and precharge time depends on ratio of latency+precharge to column access time.
Fig. 11. Multibank system bandwidth calculations.

Metal1 on top of diffusion

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.
Precharged ROM

Other Memory Cells

(a) DRAM
(b) SRAM
(c) EPROM
(d) Mask ROM

Floating gate

Cutting

NOR Array
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

- Higher Vth (around 7v) means that 5v Vgs no longer turns on the transistor
- SiO2 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

- Thin oxide allows erasing in-system
  - Fowler-Nordheim Tunneling

- 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 SiO2
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

Flash EEPROM
## Realistic PROM Devices

<table>
<thead>
<tr>
<th></th>
<th>EPROM [Tomita91]</th>
<th>EEPROM [Terada89, Pashley89]</th>
<th>Flash EEPROM [Jinbo92]</th>
</tr>
</thead>
<tbody>
<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 17.39 mm²</td>
<td>11.8 x 7.7 mm²</td>
<td>6.3 x 18.5 mm²</td>
</tr>
<tr>
<td>Cell size</td>
<td>3.8 μm²</td>
<td>30 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 msec/word, 4 sec /chip</td>
<td>5 μsec</td>
</tr>
<tr>
<td>Erase/Write cycles</td>
<td>100</td>
<td>10⁵</td>
<td>10³-10⁵</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
Add the Match line
   - Essentially a distributed NOR gate
Programmable Logic Array

AND PLANE

OR PLANE

Product Terms

x_0 x_1 x_2

f_0 f_1

PLA

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