15. Memories

- This module
  - Memory arrays
  - SRAMs
  - Serial Memories
  - Dynamic memories

### Memory Arrays

- Random Access Memory (RAM)
- Serial Access Memory
- Content Addressable Memory (CAM)
- Static RAM (SRAM)
- Dynamic RAM (DRAM)
- Shift Registers
- Queues

### Array Architecture

- $2^n$ words of $2^m$ bits each
- If $n \gg m$, fold by $2^k$ into fewer rows of more columns
- Good regularity – easy to design
- Very high density if good cells are used

### 12T SRAM Cell

- Basic building block: SRAM Cell
  - Holds one bit of information, like a latch
  - Must be read and written
- 12-transistor (12T) SRAM cell
  - Use a simple latch connected to bitline
  - 46 x 75 μm unit cell
6T SRAM Cell

- Cell size accounts for most of array size
  - Reduce cell size at expense of complexity
- 6T SRAM Cell
  - Used in most commercial chips
  - Data stored in cross-coupled inverters
- Read:
  - Precharge bit, bit\(_b\)
  - Raise wordline
- Write:
  - Drive data onto bit, bit\(_b\)
  - Raise wordline

SRAM Read

- Improve performance when bit-line capacitance is high
- Precharge both bitlines high
- Then turn on wordline
- One of the two bitlines will be pulled down by the cell
- Ex: A = 0, A\(_b\) = 1
  - bit discharges, bit\(_b\) stays high
  - But A bumps up slightly
- Read stability
  - A must not flip
  - N1 >> N2

SRAM Write

- Drive one bitline high, the other low
- Then turn on wordline
- Bitlines overpower cell with new value
- Ex: A = 0, A\(_b\) = 1,
  - bit = 1, bit\(_b\) = 0
  - Force A\(_b\) low,
  - then A rises high
- Writability
  - Must overpower feedback inverter
  - N4 >> P2
  - N2 >> P1 (symmetry)

SRAM Sizing

- High bitlines must not overpower inverters during reads
- But low bitlines must write new value into cell
15. Memories

SRAM Column Example

Read

Write

SRAM Layout

• Cell size is critical: 26 x 45 \( \lambda \) (even smaller in industry)
• Tile cells sharing \( V_{DD} \), GND, bitline contacts

Decoders

• \( n:2^n \) decoder consists of \( 2^n \) n-input AND gates
  – One needed for each row of memory
  – Build AND from NAND or NOR gates

Decoder Layout

• Decoders must be pitch-matched to SRAM cell
  – Requires very skinny gates
15. Memories

Large Decoders
- For \( n > 4 \), NAND gates become slow
  - Break large gates into multiple smaller gates

Predecoding
- Many of these gates are redundant
  - Factor out common gates into predecoder
  - Saves area
  - Same path effort

Column Circuitry
- Some circuitry is required for each column
  - Bitline conditioning
  - Sense amplifiers
  - Column multiplexing

Bitline Conditioning
- Precharge bitlines high before reads
- Equalize bitlines to minimize voltage difference when using sense amplifiers
15. Memories

Sense Amplifiers

- Bitlines have many cells attached
  - Ex: 32-kbit SRAM has 256 rows x 128 cols
  - 128 cells on each bitline
- \( t_{pd} \propto (C/I) \Delta V \)
  - Even with shared diffusion contacts, 64C of diffusion capacitance (big C)
  - Discharged slowly through small transistors (small I)
- Sense amplifiers are triggered on small voltage swing (reduce \( \Delta V \))

Differential Pair Amp

- Differential pair requires no clock
- But always dissipates static power

Clocked Sense Amp

- Clocked sense amp saves power
- Requires sense_clk after enough bitline swing
- Isolation transistors cut off large bitline capacitance

Twisted Bitlines

- Sense amplifiers also amplify noise
  - Coupling noise is severe in modern processes
  - Try to couple equally onto bit and bit_b
  - Done by twisting bitlines
Column Multiplexing

- Recall that array may be folded for good aspect ratio
- Ex: 2 kword x 16 folded into 256 rows x 128 columns
  - Must select 16 output bits from the 128 columns
  - Requires 16 8:1 column multiplexers

Tree Decoder Mux

- Column mux can use pass transistors
  - Use nMOS only, precharge outputs
- One design is to use k series transistors for $2^k$:1 mux
  - No external decoder logic needed

Single Pass-Gate Mux

- Or eliminate series transistors with separate decoder

Ex: 2-way Muxed SRAM

- More Cells
- word_q1
- write0_q1
- data_v1
- write1_q1
15. Memories

Multiple Ports

- We have considered single-ported SRAM
  - One read or one write on each cycle
- Multiported SRAM are needed for register files
- Examples:
  - Multicycle MIPS must read two sources or
    write a result on some cycles
  - Pipelined MIPS must read two sources and
    write a third result each cycle
  - Superscalar MIPS must read and write many
    sources and results each cycle

Dual-Ported SRAM

- Simple dual-ported SRAM
  - Two independent single-ended reads
  - Or one differential write
  - Do two reads and one write by time
    multiplexing
  - Read during ph1, write during ph2

Multi-Ported SRAM

- Adding more access transistors hurts read stability
- Multiported SRAM isolates reads from state node
- Single-ended design minimizes number of bitlines
15. Memories

Serial Access Memories

- Serial access memories do not use an address
  - Shift Registers
  - Tapped Delay Lines
  - Serial In Parallel Out (SIPO)
  - Parallel In Serial Out (PIPO)
  - Queues (FIFO, LIFO)

Shift Register

- Shift registers store and delay data
- Simple design: cascade of registers
  - Watch your hold times!

Denser Shift Registers

- Flip-flops aren’t very area-efficient
- For large shift registers, keep data in SRAM instead
- Move R/W pointers to RAM rather than data
  - Initialize read address to first entry, write to last
  - Increment address on each cycle
**Tapped Delay Line**

- A *tapped delay line* is a shift register with a programmable number of stages
- Set number of stages with delay controls to mux
  - Ex: 0 – 63 stages of delay

**Serial In Parallel Out**

- 1-bit shift register reads in serial data
  - After N steps, presents N-bit parallel output

**Parallel In Serial Out**

- Load all N bits in parallel when shift = 0
  - Then shift one bit out per cycle

**Queues**

- *Queues* allow data to be read and written at different rates.
- Read, Write each use their own clock, data
- Queue indicates whether it is full or empty
- Build with SRAM and read/write counters (pointers)
FIFO, LIFO Queues

- **First In First Out (FIFO)**
  - Initialize read and write pointers to first element
  - Queue is EMPTY
  - On write, increment write pointer
  - If write almost catches read, Queue is FULL
  - On read, increment read pointer

- **Last In First Out (LIFO)**
  - Also called a stack
  - Use a single stack pointer for read and write

4-Transistor Dynamic RAM Cell

Remove the two p-channel transistors from static RAM cell, to get a four-transistor dynamic RAM cell

Data stored as charge on gate capacitors (complementary nodes)

Data must be refreshed regularly

Dynamic cells must be designed very carefully

3-Transistor Dynamic RAM Cell

Data stored on the gate of a transistor

Need two additional transistors, one for write and the other for read control

1-Transistor Dynamic RAM Cell

Value of $C_B$ must be chosen very carefully; otherwise, voltage on bit-line will be affected by charge sharing

Cannot get any smaller than this: data stored on a (trench) capacitor $C$, need a transistor to control data

Bit line normally precharged to $\frac{1}{2}V_{DD}$ (need a sense amplifier)
Single-Event Upsets

Interaction of a Cosmic Ray and Silicon MOS Transistor

Particle produces electron-hole pairs in substrate; when collected at source and drain, will cause current pulse.

A "bit-flip" can occur in the memory cell due to the charge generated by the particle – called a "single-event upset".

Seen in spacecraft electronics in the past

Source: Aerospace Corporation

Karu Sankaralingam