EECS151: Introduction to Digital Design and ICs

Lecture 23 – SRAM

Bora Nikolić

Review

• Latches are based on positive feedback
• Clk-Q delay calculated similarly to combinational logic
• Setup, hold defined as D-Clk times that correspond to Clk-Q delay increases
• Flip-flop is typically a latch pair

Random Access Memory Architecture

• Conceptual: Linear array of addresses
  • Each box holds some data
  • Not practical to physically realize
    – millions of 32b/64b words
  • Create a 2-D array
    • Decode Row and Column address to get data

Basic Memory Array

• Core
  • Wordlines to access rows
  • Bitlines to access columns
  • Data multiplexed onto columns

• Decoders
  • Addresses are binary
  • Row/column MUXes are 'one-hot' - only one is active at a time

Basic Static Memory Element

• If \( D \) is high, \( D \) will be driven low
  • Which makes \( D \) stay high
  • Positive feedback
  • Same principle as in latches

Positive Feedback: Bi-Stability

• As in latches

Writing into a Cross-Coupled Pair

• This is a 5T SRAM cell
  • Access transistor must be able to overpower the feedback; therefore must be large
  • Easier to write a 0, harder to write 1
  • Can implement as a transmission gate as well; single-ended 6T cell
  • There is a better solution...
SRAM Cell

Since it is easier to write a 0 through NMOS, write only 0s, but on opposite sides! When reading, measure the difference.

6-transistor CMOS SRAM Cell

- Wordline (WL) enables read/write access for a row.
- Data is written/read differentially through shared BL, BL.

SRAM Operation

Write

Hold

SRAM read in non-destructive - Reading the cell should not destroy the stored value.

Sizing SRAM Cell

- Read stability: Cell should not change value during read.
  * Q = 0: M5, M1 both on.
  * Voltage divider between M5, M1.
  * VQ should stay low, not to flip M4-M3 inverter.
  * (W/L)M1 > (W/L)M5.
  * Typically (W/L)M1 = 1.5 (W/L)M5.
  * In finFETs: (W/L)M1 = 2(W/L)M5.
  * Read speed: Both M5 and M1.

SRAM Column: Write

SRAM Column: Read
**Administrivia**

- Homework 10 posted on Friday, due 11/22
- No homework during Thanksgiving
- Project checkpoints #3 this week

**Multi-Ported Memory**

- Motivation:
  - Consider CPU core register file:
    - 1 read or write per cycle limits processor performance.
    - Complicates pipelining. Difficult for different instructions to simultaneously read or write reg file.
    - Single-issue pipelined CPUs usually need 2 read ports and 1 write port (2R/1W).
    - Superscalar processors have more (e.g. 6R/3W).
  - I/O data buffering: Dual-port read/write allows both sides to simultaneously access memory at full bandwidth.

**Dual-Ported Memory Internals**

- Add decoder, another set of read/write logic, bits lines, word lines:
- Example cell: SRAM
- Dual-porting allows both sides to simultaneously access memory at full bandwidth.
- Repeat everything but cross-coupled inverters.
- This scheme extends up to a couple more ports, then need to add additional transistors.

**1R/1W 8T SRAM**

- Dual-port read/write capability
- Single-cycle read and write, timed appropriately
- Often found in register files, first level (L1) of cache

**True or False**

1. Transistor leakage doesn’t affect SRAM read speed
2. One should write into an SRAM cell by pulling BL high
3. One can only write into some cells of a selected WL

**Decoders**

- Intuitive architecture for N x M memory
- Too many select signals:
  - N words = N select signals
  - Decoder reduces the number of select signals
  - Intuitive architecture for N x M memory
  - To minimize select signals:
  - N words = N select signals
  - Decoder reduces the number of select signals
  - $N = \log_2 N$
Row Decoders

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

(N)AND Decoder
\[ W_{D} = A_0 \cdot A_1 \cdot A_2 \cdot A_3 \cdot A_4 \cdot A_5 \cdot A_6 \cdot A_7 \]

NOR Decoder
\[ W_{L} = \overline{A_0} \cdot \overline{A_1} \cdot \overline{A_2} \cdot \overline{A_3} \cdot \overline{A_4} \cdot \overline{A_5} \cdot \overline{A_6} \cdot \overline{A_7} \]

Possible Decoder
- 256 8-input AND gates
  - Each built out of a tree of NAND gates and inverters
- Need to drive a lot of capacitance (SRAM cells)
  - What’s the best way to do this?

Possible AND8
- Build 8-input NAND gate using 2-input gates and inverters
- Is this the best we can do?
- Is this better than using fewer NAND4 gates?

Problem Setup
- Goal: Build fastest possible decoder with static CMOS logic
- What we know
  - Basically need 256 AND gates, each one of them drives one word line

Problem Setup (1)
- Each wordline has 256 cells connected to it
- \[ C_{WL} = 256 \cdot C_{cell} + C_{wire} \]
  - Ignore wire for now
- Assume that decoder input capacitance is \[ C_{address} = 4 \cdot C_{cell} \]

Problem Setup (2)
- Each address bit drives \( 2^k/2 \) AND gates
  - \( A_0 \) drives \( 1/2 \) of the gates, \( A_0_b \) the other \( 1/2 \) of the gates

Decoder Fan-Out
- \( F \) of \( 2^{13} \) means that we will want to use more than \( \log_2(2^{13}) = 6.5 \) stages to implement the AND8
- Need many stages anyways
  - So what is the best way to implement the AND gate?
  - Will see next that it’s the one with the most stages and least complicated gates
8-Input AND

- Using 2-input NAND gates
- 8-input gate takes 6 stages
- Total G is \((3/2)^3 \approx 3.4\)
- So H is \(3.4 \times 2^{13} \approx 7.4\)

Decoder So Far

- 256 8-input AND gates
  - Each built out of tree of NAND gates and inverters
- Issue:
  - Every address line has to drive 128 gates and wire right away
  - Forces us to add buffers just to drive address inputs

Predecoders

- Use a single gate for each of the shared terms
  - E.g., from \(A_0, \overline{A}_0, A_1,\) and \(\overline{A}_1\), generate four signals: \(A_0A_1, \overline{A}_0A_1, A_0\overline{A}_1, A_0A_1\)
- In other words, we are decoding smaller groups of address bits first
  - And using the "predecoded" outputs to do the rest of the decoding

Predecode Options

- Larger predecode usually better:
- More stages before the-long wires
  - Decreases their affect on the circuit
- Fewer number of long wires switches
  - Lower power
- Easier to fit 2-input gate into cell pitch

Building Larger Arrays
Building Larger Custom Arrays

- Each subarray is 2-8kB
- Hierarchical decoding
- Peripheral overhead is 30-50%
- Delay is wire dominated
- Scratchpads, caches, TLBs

Adding Ports to Primitive Memory Blocks

How to add a write port to a simple dual-port memory.
Example: given 1Kx8 SDP, want 1 read & 2 write ports.

Cascading Memory-Blocks

How to make larger memory blocks out of smaller ones.
Increasing the width. Example: given 1Kx8, want 1Kx16

Adding Ports to Primitive Memory Blocks

How to add a read port to a simple dual-port (SDP) memory.
Example: given 1Kx8 SDP, want 1 write & 2 read ports.

Review

- Dense memories are built as arrays of memory elements
  - SRAM is a static memory
- SRAM has unique combination of density, speed, power
- SRAM cells sized for stability and writeability
- SRAM and regfile cells can have multiple R/W ports
- Memory decoding is done hierarchically
  - Wire-limited in large arrays