CMOS Bistable

- Cross-coupled inverters used to hold state in CMOS
- "Static" storage in powered cell, no refresh needed
  - If a storage node leaks or is pushed slightly away from correct value, non-linear transfer function of high-gain inverter removes noise and recirculates correct value
- To write new state, have to force nodes to opposite state
CMOS Transparent Latch

Latch transparent (output follows input) when clock is high, holds last value when clock is low

Transmission gate switch with both pMOS and nMOS passes both ones and zeros well

Schematic Symbols

Transparent on clock low
Latch Operation

Clock High
Latch Transparent

Clock Low
Latch Holding
Flip-Flop as Two Latches

This is how standard cell flip-flops are built (usually with extra in/out buffers)

Schematic Symbols

D

Q

Clk

Clk
Small Memories from Stdcell Latches

- Add additional ports by replicating read and write port logic (multiple write ports need mux in front of latch)
- Expensive to add many ports

Addition of Ports:
- Use combinatorial logic for read port (synthesized)
- Data held in transparent-low latches
- Optional read output latch

Diagrams:
- Write Address Decoder
- Write Address
- Write Data
- Read Address Decoder
- Read Address
- Clk
- Write by clocking latch
- Data held in transparent-low latches

Lecture 9, Memory
6-Transistor SRAM (Static RAM)

- Large on-chip memories built from arrays of static RAM bitcells, where each bit cell holds a bistable (cross-coupled inverters) and two access transistors.
- Other clocking and access logic factored out into periphery
Intel’s 22nm SRAM cell

- 0.092 um² SRAM cell for high density applications
- 0.108 um² SRAM cell for low voltage applications

[Bohr, Intel, Sept 2009]
General SRAM Structure

- Address
- Decode and Wordline Driver
- Bitline Prechargers
- Differential Read Sense Amplifiers
- Differential Write Drivers
- Clk
- Write Enable
- Write Data
- Read Data

Usually maximum of 128-256 bits per row or column
**Address Decoder Structure**

- **Unary 1-of-4 encoding**
- **2:4 Predecoders**
- **Clocked Word Line Enable**

The diagram shows a memory address decoder with a 4-bit address (A0 to A3) and 16 word lines (Word Line 0 to Word Line 15). The address lines are connected to predecoders that generate the word line enable signals.
Read Cycle

1) Precharge bitlines and senseamp
2) Pulse wordlines, develop bitline differential voltage
3) Disconnect bitlines from senseamp, activate sense pulldown, develop full-rail data signals

Pulses generated by internal self-timed signals, often using “replica” circuits representing critical paths
Write Cycle

1) Precharge bitlines

2) Open wordline, pull down one bitline full rail

Write-enable can be controlled on a per-bit level. If bit lines not driven during write, cell retains value (looks like a read to the cell).
Difficult to pitch match sense amp to tight SRAM bit cell spacing so often 2-8 columns share one sense amp. Impacts power dissipation as multiple bitline pairs swing for each bit read.

Lecture 9, Memory
Building Larger Memories

- Large arrays constructed by tiling multiple leaf arrays, sharing decoders and I/O circuitry
  - e.g., sense amp attached to arrays above and below
- Leaf array limited in size to 128-256 bits in row/column due to RC delay of wordlines and bitlines
- Also to reduce power by only activating selected sub-bank
- In larger memories, delay and energy dominated by I/O wiring
Adding More Ports

WordlineB
WordlineA
BitB
BitA

Differential Read or Write ports

Optional Single-ended Read port
Memory Compilers

- In ASIC flow, memory compilers used to generate layout for SRAM blocks in design
  - Often hundreds of memory instances in a modern SoC
  - Memory generators can also produce built-in self-test (BIST) logic, to speed manufacturing testing, and redundant rows/columns to improve yield

- Compiler can be parameterized by number of words, number of bits per word, desired aspect ratio, number of sub banks, degree of column muxing, etc.
  - Area, delay, and energy consumption complex function of design parameters and generation algorithm
  - Worth experimenting with design space

- Usually only single read or write port SRAM and one read and one write SRAM generators in ASIC library
Small Memories

- Compiled SRAM arrays usually have a high overhead due to peripheral circuits, BIST, redundancy.
- Small memories are usually built from latches and/or flip-flops in a stdcell flow.
- Cross-over point is usually around 1K bits of storage.
- Should try design both ways.
Memory Design Patterns
Multiport Memory Design Patterns

Often we require multiple access ports to a common memory

- True Multiport Memory
  - As described earlier in lecture, completely independent read and write port circuitry

- Banked Multiport Memory
  - Interleave lesser-ported banks to provide higher bandwidth

- Stream-Buffered Multiport Memory
  - Use single wider access port to provide multiple narrower streaming ports

- Cached Multiport Memory
  - Use large single-port main memory, but add cache to service
True Multiport Memory

**Problem:** Require simultaneous read and write access by multiple independent agents to a shared common memory.

**Solution:** Provide separate read and write ports to each bit cell for each requester

**Applicability:** Where unpredictable access latency to the shared memory cannot be tolerated.

**Consequences:** High area, energy, and delay cost for large number of ports. Must define behavior when multiple writes on same cycle to same word (e.g., prohibit, provide priority, or combine writes).
True Multiport Example: Itanium-2 Regfile

- Intel Itanium-2 [Fetzer et al, IEEE JSSCC 2002]

The integer datapath bypassing is divided into four stages, to afford more timing critical inputs the least possible logic delay to the consuming ALUs. Critical L1 cache return data must flow through only one level of muxing before arriving at the ALU inputs, while DET and WRB data, available from staging latches, have the longest logic path to the ALUs. This allows the bypassing of operands from 34 possible results to occur in a half clock cycle, enabling a single-cycle cache access and instruction execution.
A Fully Bypassed Six-Issue Integer Datapath and Register File on the Itanium-2 Microprocessor

Abstract—The six-issue integer datapath of the second-generation Itanium Microprocessor is described. Pulse techniques enable a high-speed, 20-ported, 128-entry, 65-bit register file with only 12 wordlines per register. A four-stage operand bypass network achieves a fully bypassed design with operands sourced from 34 locations with 16 destinations. To control this network, over 280 bypass comparators are utilized. Using half a clock for execution and half a clock for bypass, each result is available for the next instruction. Functional units are pre-enabled, reducing power consumption by 15% while eliminating a stage of result muxing and improving performance. The part is fabricated in a six-layer, 18-m process and operates at 1.0 GHz at 1.5 V, consuming less than 130 W in about 420 mm.

I. INTRODUCTION

The Itanium-2 microprocessor, the second implementation of the Itanium architecture, features an explicitly parallel architecture. This architecture lends itself to a highly superscalar implementation. Highly superscalar designs require larger register files (RFs) to feed multiple execution units and complex bypass networks to keep data freely moving through the system. The Itanium-2 microprocessor incorporates a six-issue integer datapath (IEU) with a 20-ported, 128-entry, 65-bit-wide RFs. To prevent data hazards, integer operands are fully bypassed through four stages of bypass multiplexing with each of the 12 integer operands and four data cache addresses sourced from 34 possible results (RF, instruction field, two L1 data caches, six arithmetic and logic (ALU) EXE stages (Fig. 1), eight DET stages, eight WRB-stage integers, six WRB-stage multimedia, and various architected registers).

II. DATAPATH

A. RF

The 20-ported RF is 2.2 mm and incorporates 12 read and eight write ports. To accomplish all required bypassing, RF reads occur simultaneously with the first two stages of bypass and ALU execution (Fig. 1). As the RF read is dumped onto the read data bit line, the RF word line decoder is sent the address for the write. During the read, the write data is also bypassed along the write bit lines reusing these bit lines for a bypass from the six-issue multimedia unit to the integer datapath. Single-ended read performance is maintained with a two-level bit line structure saving wires and reducing delay (by removing self-timed path margin) relative to sense-amp designs of the past [1], [2]. Registers are banked in arrays of 16 with bit lines in the second metal layer. The bit lines are then repeated and promoted to the fourth metal layer to drive the outputs to the middle bypass mux in the integer and multimedia units (MMU).

In a standard register file, 20 ports would require 20 word lines per register. Such a design would have exceeded area and timing constraints. Double pumping the word lines achieves same cycle reads and writes with a single wire. Each register has only 12 word lines and a shared control wire (WRITEH). WRITEH determines the read/write directionality of the word line with selection internal to each register (Fig. 2). The word line decoder decodes read and write register addresses. To do this the decoder is clocked with a pulse on the rising and falling edge of the clock. The register address is statically decoded with the results of the high- and low-order bits being strobed by the double pumped pulse clock (PCK2X) generated by the circuit shown in Fig. 3. Pulses do not propagate well through many
Banked Multiport Memory

**Problem:** Require simultaneous read and write access by multiple independent agents to a large shared common memory.

**Solution:** Divide memory capacity into smaller banks, each of which has fewer ports. Requests are distributed across banks using a fixed hashing scheme. Multiple requesters arbitrate for access to same bank/port.

**Applicability:** Requesters can tolerate variable latency for accesses. Accesses are distributed across address space so as to avoid “hotspots”.

**Consequences:** Requesters must wait arbitration delay to determine if request will complete. Have to provide interconnect between each requester and each bank/port. Can have greater, equal, or lesser number of banks*ports/bank compared to total number of external access ports.
Banked Multiport Memory

Port A

Bank 0

Bank 1

Bank 2

Bank 3

Arbitration and Crossbar

Port B
Banked Multiport Memory Example

Pentium (P5) 8-way interleaved data cache, with two ports

Figure 7. Dual-access data cache.

[Alpert et al, IEEE Micro, May 1993]
Stream-Buffered Multiport Memory

**Problem:** Require simultaneous read and write access by multiple independent agents to a large shared common memory, where each requester usually makes multiple sequential accesses.

**Solution:** Organize memory to have a single wide port. Provide each requester with an internal stream buffer that holds width of data returned/consumed by each memory access. Each requester can access own stream buffer without contention, but arbitrates with others to read/write stream buffer from memory.

**Applicability:** Requesters make mostly sequential requests and can tolerate variable latency for accesses.

**Consequences:** Requesters must wait arbitration delay to determine if request will complete. Have to provide stream buffers for each requester. Need sufficient access width to serve aggregate bandwidth demands of all requesters, but wide data access can be wasted if not all used by requester. Have to specify memory consistency model between ports (e.g., provide stream flush operations).
Stream-Buffered Multiport Memory

Stream Buffer A

Stream Buffer B

Arbitration

Wide Memory

Port A

Port B
Stream-Buffered Multiport Examples

- IBM Cell microprocessor local store

[Chen et al., IBM, 2005]
Cached Multiport Memory

**Problem:** Require simultaneous read and write access by multiple independent agents to a large shared common memory.

**Solution:** Provide each access port with a local cache of recently touched addresses from common memory, and use a cache coherence protocol to keep the cache contents in sync.

**Applicability:** Request streams have significant temporal locality, and limited communication between different ports.

**Consequences:** Requesters will experience variable delay depending on access pattern and operation of cache coherence protocol. Tag overhead in both area, delay, and energy/access. Complexity of cache coherence protocol.
Cached Multiport Memory

Port A

Cache A

↑

↓

Arbitration and Interconnect

↓

Common Memory

Port B

Cache B

↑

↓
Replicated-State Multiport Memory

**Problem:** Require simultaneous read and write access by multiple independent agents to a small shared common memory. Cannot tolerate variable latency of access.

**Solution:** Replicate storage and divide read ports among replicas. Each replica has enough write ports to keep all replicas in sync.

**Applicability:** Many read ports required, and variable latency cannot be tolerated.

**Consequences:** Potential increase in latency between some writers and some readers.
Replicated-State Multiport Memory

Write Port 0  Write Port 1

Copy 0  Copy 1

Read Ports

Example: Alpha 21264 Regfile clusters
Memory Hierarchy Design Patterns

Use small fast memory together large slow memory to provide illusion of large fast memory.

- Explicitly managed local stores
- Automatically managed cache hierarchies