Dynamic Scheduling:
- Scoreboarding/Tomasulo
- In-order issue, out-of-order execution, out-of-order commit

Register renaming:
- Replaces registers names from original code (external names) with pointers to internal registers
- Removes WAR and WAW hazards, since each write to a register picks a new mapping between external and internal names.

Branch prediction/speculation
- Regularities in program execution permit prediction of branch directions and data values
- Necessary for wide superscalar issue

Reorder Buffer
- Provides in-order-commit
- Precise exceptions/recovery from mis-prediction

Review: advanced pipelining

The Five Classic Components of a Computer

Today's Topics:
- Recap last lecture
- Locality and Memory Hierarchy
- Administrivia
- SRAM Memory Technology
- DRAM Memory Technology
- Memory Organization

The Big Picture: Where are We Now?

Technology Trends (from 1st lecture)

<table>
<thead>
<tr>
<th>Capacity</th>
<th>Speed (latency)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic:</td>
<td>2x in 3 years</td>
</tr>
<tr>
<td>DRAM:</td>
<td>4x in 3 years</td>
</tr>
<tr>
<td>Disk:</td>
<td>4x in 3 years</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DRAM Capacity</th>
<th>Cycle Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1980 64 Kb</td>
<td>250 ns</td>
</tr>
<tr>
<td>1983 256 Kb</td>
<td>220 ns</td>
</tr>
<tr>
<td>1986 1 Mb</td>
<td>190 ns</td>
</tr>
<tr>
<td>1989 4 Mb</td>
<td>165 ns</td>
</tr>
<tr>
<td>1992 16 Mb</td>
<td>145 ns</td>
</tr>
<tr>
<td>1995 64 Mb</td>
<td>120 ns</td>
</tr>
</tbody>
</table>
Who Cares About the Memory Hierarchy?

Processor-DRAM Memory Gap (latency)

- **µProc**: 60%/yr. (2X/1.5yr)
- **DRAM**: 9%/yr. (2X/10 yrs)

**“Moore’s Law”**

Processor-Memory Performance Gap: (grows 50% / year)

Time

Impact on Performance

- Suppose a processor executes at
  - Clock Rate = 200 MHz (5 ns per cycle)
  - CPI = 1.1
  - 50% arith/logic, 30% ld/st, 20% control

- Suppose that 10% of memory operations get 50 cycle miss penalty

- CPI = ideal CPI + average stalls per instruction
  = \(1.1 \times 0.30 + 0.10 \times 50\) (cycle/miss)
  = 2.6

- 58% of the time the processor is stalled waiting for memory!

- a 1% instruction miss rate would add an additional 0.5 cycles to the CPI!

Today’s Situation: Microprocessor

- Rely on caches to bridge gap

  - **Microprocessor-DRAM performance gap**
    - time of a full cache miss in instructions executed
      - 1st Alpha (7000): 340 ns/5.0 ns = 68 clks x 2 or 136 instructions
      - 2nd Alpha (8400): 266 ns/3.3 ns = 80 clks x 4 or 320 instructions
      - 3rd Alpha (t.b.d.): 180 ns/1.7 ns = 108 clks x 6 or 648 instructions
    - 1/2X latency x 3X clock rate x 3X Instr/clock => -5X

The Goal: illusion of large, fast, cheap memory

- **Fact**: Large memories are slow, fast memories are small
- How do we create a memory that is large, cheap and fast (most of the time)?
  - Hierarchy
  - Parallelism
An Expanded View of the Memory System

<table>
<thead>
<tr>
<th>Processor</th>
<th>Control</th>
<th>Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td>Datapath</td>
<td>Memory</td>
<td>Memory</td>
</tr>
</tbody>
</table>

- **Speed:** Fastest
- **Size:** Smallest
- **Cost:** Highest
- **Slowest**
- **Biggest**
- **Lowest**

Why hierarchy works

- **The Principle of Locality:**
  - Program access a relatively small portion of the address space at any instant of time.

![Probability of reference graph](image)

Memory Hierarchy: How Does it Work?

- **Temporal Locality** (Locality in Time):
  - Keep most recently accessed data items closer to the processor

- **Spatial Locality** (Locality in Space):
  - Move blocks consists of contiguous words to the upper levels

Memory Hierarchy: Terminology

- **Hit:** data appears in some block in the upper level (example: Block X)
  - **Hit Rate:** the fraction of memory access found in the upper level
  - **Hit Time:** Time to access the upper level which consists of RAM access time + Time to determine hit/miss

- **Miss:** data needs to be retrieve from a block in the lower level (Block Y)
  - **Miss Rate** = 1 - (Hit Rate)
  - **Miss Penalty:** Time to replace a block in the upper level + Time to deliver the block to the processor

**Hit Time << Miss Penalty**
**Memory Hierarchy of a Modern Computer System**

By taking advantage of the principle of locality:

- Present the user with as much memory as is available in the cheapest technology.
- Provide access at the speed offered by the fastest technology.

![Diagram of memory hierarchy](image)

**How is the hierarchy managed?**

- **Registers <-> Memory**
  - by compiler (programmer?)

- **cache <-> memory**
  - by the hardware

- **memory <-> disks**
  - by the hardware and operating system (virtual memory)
  - by the programmer (files)

**Memory Hierarchy Technology**

- **Random Access:**
  - “Random” is good: access time is the same for all locations
  - **DRAM:** Dynamic Random Access Memory
    - High density, low power, cheap, slow
    - Dynamic: need to be “refreshed” regularly
  - **SRAM:** Static Random Access Memory
    - Low density, high power, expensive, fast
    - Static: content will last “forever” (until lose power)

- **“Non-so-random” Access Technology:**
  - Access time varies from location to location and from time to time
  - Examples: Disk, CDROM

- **Sequential Access Technology:** access time linear in location (e.g., Tape)

- The next two lectures will concentrate on random access technology

  - The Main Memory: DRAMs + Caches: SRAMs

**Main Memory Background**

- **Performance of Main Memory:**
  - Latency: Cache Miss Penalty
    - **Access Time:** time between request and word arrives
    - **Cycle Time:** time between requests
  - Bandwidth: I/O & Large Block Miss Penalty (L2)

- **Main Memory is DRAM:** Dynamic Random Access Memory
  - Dynamic since needs to be refreshed periodically (8 ms)
  - Addresses divided into 2 halves (Memory as a 2D matrix):
    - RAS or **Row Access Strobe**
    - CAS or **Column Access Strobe**

- **Cache uses SRAM:** Static Random Access Memory
  - No refresh (6 transistors/bit vs. 1 transistor)
    - **Size:** DRAM/SRAM - 4-8
    - **Cost/Cycle time:** SRAM/DRAM - 8-16
Random Access Memory (RAM) Technology

Why do computer designers need to know about RAM technology?

- Processor performance is usually limited by memory bandwidth
- As IC densities increase, lots of memory will fit on processor chip
  - Tailor on-chip memory to specific needs
    - Instruction cache
    - Data cache
    - Write buffer

What makes RAM different from a bunch of flip-flops?

- Density: RAM is much denser

Administrative Issues

- Due tonight by midnight: evaluations (problem 0 of lab 6)
- Due Wednesday: breakdown of lab 6
- Start reading Chapter 7 of your book (Memory Hierarchy)
- Second midterm coming up (Wed, April 21)
  - Microcoding/implementation of complex instructions
  - Pipelining
    - Hazards, branches, forwarding, CPI calculations
    - (may include something on dynamic scheduling)
  - Memory Hierarchy
  - Possibly something on I/O (see where we get in lectures)

Static RAM Cell

6-Transistor SRAM Cell

- Write:
  1. Drive bit lines (bit=1, bit=0)
  2. Select row

- Read:
  1. Precharge bit and bit to Vdd or Vdd/2 => make sure equal!
  2. Select row
  3. Cell pulls one line low
  4. Sense amp on column detects difference between bit and bit

Typical SRAM Organization: 16-word x 4-bit

Q: Which is longer: word line or bit line?
° Write Enable is usually active low (WE_L)

° Din and Dout are combined to save pins:
  • A new control signal, output enable (OE_L) is needed
  • WE_L is asserted (Low), OE_L is disasserted (High)
    - D serves as the data input pin
  • WE_L is disasserted (High), OE_L is asserted (Low)
    - D is the data output pin
  • Both WE_L and OE_L are asserted:
    - Result is unknown. Don’t do that!!!

° Although could change VHDL to do what desire, must do the best with what you’ve got (vs. what you need)
Classical DRAM Organization (square)

- Row and Column Address together:
  - Select 1 bit a time

- Each intersection represents a 1-T DRAM Cell

- Row and Column Address together:
  - Square root of bits per RAS/CAS

DRAM logical organization (4 Mbit)

- Column Decoder
- Sense Amps & I/O
- Memory Array (2,048 x 2,048)
- Storage
- Word Line

DRAM physical organization (4 Mbit)

- Block Row Dec. 9 : 512
- Block Row Dec. 9 : 512
- Block Row Dec. 9 : 512

Memory Systems

- address
- DRAM Controller
- Memory Timing Controller
- Bus Drivers

Tc = Tcycle + Tcontroller + Tdriver
Control Signals (RAS_L, CAS_L, WE_L, OE_L) are all active low.

Din and Dout are combined (D):
- WE_L is asserted (Low), OE_L is disasserted (High)
  - D serves as the data input pin
- WE_L is disasserted (High), OE_L is asserted (Low)
  - D is the data output pin

Row and column addresses share the same pins (A):
- RAS_L goes low: Pins A are latched in as row address
- CAS_L goes low: Pins A are latched in as column address
- RAS/CAS edge-sensitive

Key DRAM Timing Parameters

- $t_{RAC}$: minimum time from RAS line falling to the valid data output.
  - Quoted as the speed of a DRAM
  - A fast 4Mb DRAM $t_{RAC} = 60$ ns

- $t_{RC}$: minimum time from the start of one row access to the start of the next.
  - $t_{RC} = 110$ ns for a 4Mbit DRAM with a $t_{RAC}$ of 60 ns

- $t_{CAC}$: minimum time from CAS line falling to valid data output.
  - 15 ns for a 4Mbit DRAM with a $t_{RAC}$ of 60 ns

- $t_{PC}$: minimum time from the start of one column access to the start of the next.
  - 35 ns for a 4Mbit DRAM with a $t_{RAC}$ of 60 ns

These times do not include the time to drive the addresses off the microprocessor nor the memory controller overhead.

- Drive parallel DRAMs, external memory controller, bus to turn around, SIMM module, pins...
- 180 ns to 250 ns latency from processor to memory is good for a “60 ns” ($t_{RAC}$) DRAM

DRAM Performance

- A 60 ns ($t_{RAC}$) DRAM can:
  - perform a row access only every 110 ns ($t_{RC}$)
  - perform column access ($t_{CAC}$) in 15 ns, but time between column accesses is at least 35 ns ($t_{PC}$).
    - In practice, external address delays and turning around buses make it 40 to 50 ns

DRAM Write Timing

- Every DRAM access begins at:
  - The assertion of the RAS_L
  - 2 ways to write: early or late v. CAS

- Early Wr Cycle: WE_L asserted before CAS_L
- Late Wr Cycle: WE_L asserted after CAS_L

A 256K x 8 DRAM
### DRAM Read Timing

- Every DRAM access begins at:
  - The assertion of the RAS_L
  - 2 ways to read: early or late v. CAS

### Main Memory Performance

- Access Pattern without Interleaving:
  - D1 available
  - Start Access for D1
  - Start Access for D2

- Access Pattern with 4-way Interleaving:
  - CPU
  - Memory Bank 0
  - Memory Bank 1
  - Memory Bank 2
  - Memory Bank 3

- We can Access Bank 0 again

### Increasing Bandwidth - Interleaving

- DRAM (Read/Write) Cycle Time >> DRAM (Read/Write) Access Time
  - - 2:1; why?

- DRAM (Read/Write) Cycle Time:
  - How frequent can you initiate an access?
  - Analogy: A little kid can only ask his father for money on Saturday

- DRAM (Read/Write) Access Time:
  - How quickly will you get what you want once you initiate an access?
  - Analogy: As soon as he asks, his father will give him the money

- DRAM Bandwidth Limitation analogy:
  - What happens if he runs out of money on Wednesday?
Main Memory Performance

- **Timing model**
  - 1 to send address,
  - 4 for access time, 10 cycle time, 1 to send data
  - Cache Block is 4 words
- **Simple M.P.** = 4 x (1+10+1) = 48
- **Wide M.P.** = 1 + 10 + 1 = 12
- **Interleaved M.P.** = 1+10+1 + 3 =15

<table>
<thead>
<tr>
<th>address</th>
<th>address</th>
<th>address</th>
<th>address</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>49</td>
<td>50</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>53</td>
<td>54</td>
</tr>
<tr>
<td>8</td>
<td>9</td>
<td>57</td>
<td>58</td>
</tr>
<tr>
<td>12</td>
<td>13</td>
<td>61</td>
<td>62</td>
</tr>
</tbody>
</table>

Bank 0 Bank 1 Bank 2 Bank 3

Independent Memory Banks

- **How many banks?**
  - number banks number clocks to access word in bank
  - For sequential accesses, otherwise will return to original bank before it has next word ready
- **Increasing DRAM => fewer chips => harder to have banks**
  - Growth bits/chip DRAM : 50%-60%/yr
  - Nathan Myrvold M/S: mature software growth
    - (33%/yr for NT) - growth MB/$ of DRAM (25%-30%/yr)

Fewer DRAMs/System over Time

(from Pete MacWilliams, Intel)

<table>
<thead>
<tr>
<th>DRAM Generation '86 '89 '92 '96 '99 '02</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Mb 4 Mb 16 Mb 64 Mb 256 Mb 1 Gb</td>
</tr>
</tbody>
</table>

| Minimum PC Memory Size 4 MB 8 MB 16 MB 32 MB 64 MB 128 MB 256 MB |
|------------------------|------------------|------------------|------------------|------------------|------------------|
| Memory per DRAM growth | 32          | 16              | 8               | 4                | 8                | 4                |
| N rows x N column x M-bit | Read & Write M-bit at a time | Each M-bit access requires a RAS / CAS cycle |
| Memory per System growth | 8          | 4               | 2               | 1                |
| @ 25%-30% / year | @ 60% / year |

Page Mode DRAM: Motivation

- **Regular DRAM Organization:**
  - N rows x N column x M-bit
  - Read & Write M-bit at a time
  - Each M-bit access requires a RAS / CAS cycle
- **Fast Page Mode DRAM**
  - N x M “register” to save a row

- **Column Address**
  - N cols
  - N rows
  - M bits
  - M-bit Output
  - RAS_L
  - CAS_L
  - A
  - Row Address
  - Col Address
  - Junk
  - 1st M-bit Access
  - 2nd M-bit Access

- **Row Address**
  - Col Address
  - Junk
  - Col Address
  - Junk
Fast Page Mode Operation

- **Fast Page Mode DRAM**
  - \( N \times M \) “SRAM” to save a row
- **After a row is read into the register**
  - Only CAS is needed to access other M-bit blocks on that row
  - RAS_L remains asserted while CAS_L is toggled

Standards pinout, package, refresh rate, capacity, ...  
binary compatibility, IEEE 754, I/O bus

Sources Multiple Single  
Figures 1) capacity, 1a) $/bit  
 2) BW, 3) latency  
 2) cost
Improve 1) 60%, 1a) 25%,  
 2) 20%, 3) 7%  
 2) little change

Standards pinout, package, refresh rate, capacity, ...  
binary compatibility, IEEE 754, I/O bus

Sources Multiple Single  
Figures 1) capacity, 1a) $/bit  
 2) BW, 3) latency  
 2) cost
Improve 1) 60%, 1a) 25%,  
 2) 20%, 3) 7%  
 2) little change

- **Reduce cell size 2.5, increase die size 1.5**
- **Sell 10% of a single DRAM generation**
  - 6.25 billion DRAMs sold in 1996
- **3 phases: engineering samples, first customer ship(FCS), mass production**
  - Fastest to FCS, mass production wins share
- **Die size, testing time, yield => profit**
  - Yield >> 60%  
    (redundant rows/columns to repair flaws)

- **DRAM Design Goals**
  - **DRAMs: capacity +60%/yr, cost –30%/yr**
    - 2.5X cells/area, 1.5X die size in -3 years
  - **1997 DRAM fab line costs $1B to $2B**
    - DRAM only: density, leakage v. speed
  - **Rely on increasing no. of computers & memory per computer (60% market)**
    - SiMM or DIMM is replaceable unit  
      => computers use any generation DRAM
  - **Commodity, second source industry**
    => high volume, low profit, conservative
    - Little organization innovation in 20 years  
      page mode, EDO, Synch DRAM
  - **Order of importance: 1) Cost/bit 1a) Capacity**
    - RAMBUS: 10X BW, +30% cost => little impact
Today's Situation: DRAM

- Commodity, second source industry
  - high volume, low profit, conservative
    - Little organization innovation (vs. processors)
      in 20 years: page mode, EDO, Synch DRAM

- DRAM industry at a crossroads:
  - Fewer DRAMs per computer over time
    - Growth bits/chip DRAM: 50%-60%/yr
    - Nathan Myrvold M/S: mature software growth
      (33%/yr for NT) - growth MB/$ of DRAM (25%-30%/yr)
  - Starting to question buying larger DRAMs?

Today’s Situation: DRAM

• Intel: 30%/year since 1987; 1/3 income profit

Summary:

- Two Different Types of Locality:
  - Temporal Locality (Locality in Time): If an item is referenced, it will tend to be referenced again soon.
  - Spatial Locality (Locality in Space): If an item is referenced, items whose addresses are close by tend to be referenced soon.

- By taking advantage of the principle of locality:
  - Present the user with as much memory as is available in the cheapest technology.
  - Provide access at the speed offered by the fastest technology.

- DRAM is slow but cheap and dense:
  - Good choice for presenting the user with a BIG memory system

- SRAM is fast but expensive and not very dense:
  - Good choice for providing the user FAST access time.

Summary: Processor-Memory Performance Gap “Tax”

<table>
<thead>
<tr>
<th>Processor</th>
<th>% Area</th>
<th>% Transistors</th>
<th>(-cost)</th>
<th>(-power)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alpha 21164</td>
<td>37%</td>
<td>77%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>StrongArm SA110</td>
<td>61%</td>
<td>94%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pentium Pro</td>
<td>64%</td>
<td>88%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- 2 dies per package: Proc/I$/D$ + L2$
- Caches have no inherent value, only try to close performance gap