CH05 Internal Memory

- Computer Memory System Overview
- Semiconductor Main Memory
- Cache Memory
- Pentium II and PowerPC Cache Organizations
- Advanced DRAM Organization

Memory Hierarchy

Characteristics

- Location
- Capacity
- Unit of transfer
- Access method
- Performance
- Physical type
- Physical characteristics
- Organisation

Location

- CPU
- Internal
- External

Capacity

- Word size
  - The natural unit of organisation
- Number of words
  - or Bytes

Unit of Transfer

- Internal
  - Usually governed by data bus width
- External
  - Usually a block which is much larger than a word
  - Addressable unit
    - Smallest location which can be uniquely addressed
    - Word internally
    - Cluster on disks
Access Methods (1)

- Sequential
  - Start at the beginning and read through in order
  - Access time depends on location of data and previous location
  - e.g. tape

- Direct
  - Individual blocks have unique address
  - Access is by jumping to vicinity plus sequential search
  - Access time depends on location and previous location
  - e.g. disk

Access Methods (2)

- Random
  - Individual addresses identify locations exactly
  - Access time is independent of location or previous access
  - e.g. RAM

- Associative
  - Data is located by a comparison with contents of a portion of the store
  - Access time is independent of location or previous access
  - e.g. cache

Memory Hierarchy

- Registers
  - In CPU

- Internal or Main memory
  - May include one or more levels of cache
  - “RAM”

- External memory
  - Backing store

Performance

- Access time
  - Time between presenting the address and getting the valid data

- Memory Cycle time
  - Time may be required for the memory to “recover” before next access
  - Cycle time is access + recovery

- Transfer Rate
  - Rate at which data can be moved

Physical Types

- Semiconductor
  - RAM

- Magnetic
  - Disk & Tape

- Optical
  - CD & DVD

- Others
  - Bubble
  - Hologram

Physical Characteristics

- Decay
- Volatility
- Erasable
- Power consumption
**Organisation**
- Physical arrangement of bits into words
- Not always obvious
- e.g. interleaved

**The Bottom Line**
- How much?
  - Capacity
- How fast?
  - Time is money
- How expensive?

**Hierarchy List**
- Registers
- L1 Cache
- L2 Cache
- Main memory
- Disk cache
- Disk
- Optical
- Tape

**So you want fast?**
- It is possible to build a computer which uses only static RAM (see later)
- This would be very fast
- This would need no cache
  - How can you cache cache?
- This would cost a very large amount

**Semiconductor Memory**
- RAM
  - Misnamed as all semiconductor memory is random access
  - Read/Write
  - Volatile
  - Temporary storage
  - Static or dynamic

**Dynamic RAM**
- Bits stored as charge in capacitors
- Charges leak
- Need refreshing even when powered
- Simpler construction
- Smaller per bit
- Less expensive
- Need refresh circuits
- Slower
- Main memory
**Static RAM**
- Bits stored as on/off switches
- No charges to leak
- No refreshing needed when powered
- More complex construction
- Larger per bit
- More expensive
- Does not need refresh circuits
- Faster
- Cache

**Read Only Memory (ROM)**
- Permanent storage
- Microprogramming (see later)
- Library subroutines
- Systems programs (BIOS)
- Function tables

**Types of ROM**
- Written during manufacture
  - Very expensive for small runs
  - Programmable (once)
    - PROM
    - Needs special equipment to program
  - Read “mostly”
    - Erasable Programmable (EPROM)
      - Erased by UV
    - Electrically Erasable (EEPROM)
      - Takes much longer to write than read
  - Flash memory
    - Erase whole memory electrically

**Organisation in detail**
- A 16Mbit chip can be organised as 1M of 16 bit words
- A bit per chip system has 16 lots of 1Mbit chip with bit 1 of each word in chip 1 and so on
- A 16Mbit chip can be organised as a 2048 x 2048 x 4bit array
  - Reduces number of address pins
    - Multiplex row address and column address
    - 11 pins to address (2^11=2048)
    - Adding one more pin doubles range of values so x4 capacity

**Typical 16 Mb DRAM (4M x 4)**

**Refreshing**
- Refresh circuit included on chip
- Disable chip
- Count through rows
- Read & Write back
- Takes time
- Slows down apparent performance
Locality of Reference

- During the course of the execution of a program, memory references tend to cluster
- e.g. loops

Cache

- Small amount of fast memory
- Sits between normal main memory and CPU
- May be located on CPU chip or module
Cache operation - overview
- CPU requests contents of memory location
- Check cache for this data
- If present, get from cache (fast)
- If not present, read required block from main memory to cache
- Then deliver from cache to CPU
- Cache includes tags to identify which block of main memory is in each cache slot

Cache Design
- Size
- Mapping Function
- Replacement Algorithm
- Write Policy
- Block Size
- Number of Caches

Size does matter
- Cost
  - More cache is expensive
- Speed
  - More cache is faster (up to a point)
  - Checking cache for data takes time

Typical Cache Organization

Direct Mapping Example

<table>
<thead>
<tr>
<th>Line</th>
<th>0</th>
<th>1</th>
<th>m-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data</td>
<td>0</td>
<td>1</td>
<td>m-1</td>
</tr>
<tr>
<td>Main Memory blocks held</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0, m, 2m, 3m…2^-m</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1,m+1, 2m+1…2^-m+1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>m-1</td>
<td>m-1, 2m-1,3m-1…2^-1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Direct Mapping Cache Line Table
**Mapping Function**

- Cache of 64kByte
- Cache block of 4 bytes
  - i.e. cache is 16k (2^14) lines of 4 bytes
- 16MBytes main memory
- 24 bit address
  - (2^24=16M)

**Direct Mapping**

- Each block of main memory maps to only one cache line
  - i.e. if a block is in cache, it must be in one specific place
- Address is in two parts
- Least Significant w bits identify unique word
- Most Significant s bits specify one memory block
- The MSBs are split into a cache line field r and a tag of s-r (most significant)

**Direct Mapping Address Structure**

<table>
<thead>
<tr>
<th>Tag s-r</th>
<th>Line or Slot r</th>
<th>Word w</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>14</td>
<td>2</td>
</tr>
</tbody>
</table>

- 24 bit address
- 2 bit word identifier (4 byte block)
- 22 bit block identifier
  - 8 bit tag (=22-14)
  - 14 bit slot or line
- No two blocks in the same line have the same Tag field
- Check contents of cache by finding line and checking Tag

**Direct Mapping pros & cons**

- Simple
- Inexpensive
- Fixed location for given block
  - If a program accesses 2 blocks that map to the same line repeatedly, cache misses are very high

**Direct Mapping Cache Organization**

**Associative Mapping**

- A main memory block can load into any line of cache
- Memory address is interpreted as tag and word
- Tag uniquely identifies block of memory
- Every line’s tag is examined for a match
- Cache searching gets expensive
**Fully Associative Cache Organization**

- Each 32-bit block of data is stored with a 22-bit tag.
- The tag is compared with the tag entry in the cache to check for a hit.
- The least significant 2 bits of the address identify which byte (8-bit) is required from the 32-bit data block.

**Address Structure**

<table>
<thead>
<tr>
<th>Tag</th>
<th>Word</th>
</tr>
</thead>
<tbody>
<tr>
<td>22 bit</td>
<td>2 bit</td>
</tr>
</tbody>
</table>

- 22-bit tag stored with each 32-bit block of data.
- Compare tag field with tag entry in cache to check for a hit.
- Least significant 2 bits of address identify which byte (8-bit) is required from the 32-bit data block.

**Example**

- Address: FFFC
- Tag: FFC
- Data: 24682468
- Cache line: 3FFF

**Set Associative Mapping**

- Cache is divided into a number of sets.
- Each set contains a number of lines.
- A given block maps to any line in a given set:
  - e.g., Block B can be in any line of set 1.
  - e.g., 2 lines per set.
  - 2-way associative mapping.
  - A given block can be in one of 2 lines in only one set.

**Two Way Set Associative Mapping Example**

**Set Associative Cache Organization**
Set Associative Mapping
Address Structure

<table>
<thead>
<tr>
<th>Tag (9 bit)</th>
<th>Set (13 bit)</th>
<th>Word (2 bit)</th>
</tr>
</thead>
</table>

- Use set field to determine cache set to look in
- Compare tag field to see if we have a hit
- e.g.
  - Address | Tag | Data | Set number |
  - FF 7FF | FF | 12345678 | 1FF |
  - 001 7FFC | 001 | 11223444 | 1FF |

Replacement Algorithms (1)
Direct mapping

- No choice
- Each block only maps to one line
- Replace that line

Replacement Algorithms (2)
Associative & Set Associative

- Hardware implemented algorithm (speed)
- Least Recently used (LRU)
- e.g. in 2 way set associative
  - Which of the 2 block is LRU?
- First in first out (FIFO)
  - replace block that has been in cache longest
- Least frequently used
  - replace block which has had fewest hits
- Random

Write Policy

- Must not overwrite a cache block unless main memory is up to date
- Multiple CPUs may have individual caches
- I/O may address main memory directly

Write-through

- All writes go to main memory as well as cache
- Multiple CPUs can monitor main memory traffic to keep local (to CPU) cache up to date
- Lots of traffic
- Slows down writes

Write-back

- Updates initially made in cache only
- Update bit for cache slot is set when update occurs
- If block is to be replaced, write to main memory only if update bit is set
- Other caches get out of sync
- I/O must access main memory through cache
- N.B. 15% of memory references are writes
Newer RAM Technology

- Synchronous DRAM (SDRAM)
  - Currently on DIMMs
  - Access is synchronized with an external clock
  - Address is presented to RAM
  - RAM finds data (CPU waits in conventional DRAM)
  - Since SDRAM moves data in time with system clock, CPU knows when data will be ready
  - CPU does not have to wait, it can do something else
  - Burst mode allows SDRAM to set up stream of data and fire it out in block