Modules and connections
Components and connections

- The CU and the ALU constitute the Central Processing Unit
- Data and instructions need to get into the system and results out
  - Input/output
- Temporary storage of code, data and results is needed
  - Main memory
- All the units must be connected
- Different type of connection for different type of unit
  - Memory
  - Input/Output
  - CPU
Memory Connection

- Receives and sends data
- Receives addresses (of locations)
- Receives control signals
  - Read
  - Write
  - Timing
Input/Output Connection

- **Output**
  - Receive data from computer
  - Send data to peripheral

- **Input**
  - Receive data from peripheral
  - Send data to computer
  - Receive control signals from computer
  - Send control signals to peripherals
  - Receive addresses from computer
    - e.g. port number to identify peripheral
  - Send interrupt signals (control)
CPU Connection

- Reads instruction and data
- Writes out data (after processing)
- Sends control signals to other units
- Receives (& acts on) interrupts
Bus
Buses

- There are a number of possible interconnection systems
- Single and multiple BUS structures are most common

- A Bus is a communication pathway connecting two or more devices
- Usually broadcast
- Often grouped
  - A number of channels in one bus
  - e.g. 32 bit data bus is 32 separate single bit channels
- Power lines may not be shown
Buses Data Bus

- **Data Bus** - Carries data
  - there is no difference between “data” and “instruction”
  - Width is a key determinant of performance (8, 16, 32, 64 bit)

- **Address Bus** - Identify the source or destination of data
  - e.g. CPU needs to read an instruction (data) from a given location in memory
  - Bus width determines maximum memory capacity of system

- **Control Bus** - Control and timing information
  - Memory read/write signal
  - Interrupt request
  - Clock signals
Bus Interconnection Scheme
Physical Realization of Bus Architecture
Single Bus Problems

- Lots of devices on one bus leads to:
  - Propagation delays
    - Co-ordination of bus use can affect performance (long data paths)
    - If aggregate data transfer approaches bus capacity
  - Most systems use multiple buses to overcome these problems
High Performance Bus
Memory Hierarchy
Memory Hierarchy

- Registers
  - In CPU
- Internal or Main memory
  - More levels of cache
  - “RAM”
- External memory
  - Backing store
Characteristics

- **Location**
  - CPU
  - Internal
  - External

- **Capacity**
  - Word size
  - Number of words

- **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 uniquely addressed
Characteristics

- **Access method**
  - **Sequential (e.g. tape)**
    - Start at the beginning and read through in order
    - Access time depends on location of data and previous location
  - **Direct (e.g. disk)**
    - Individual blocks have unique address
    - Access is by jumping to vicinity plus sequential search
    - Access time depends on location and previous location
Characteristics

- **Access method**
  - **Random (e.g. RAM)**
    - Individual addresses identify locations exactly
    - Access time is independent of location or previous access
  - **Associative**
    - Data is located by a comparison with contents of a portion of the store
    - Access time is independent of location or previous access
Characteristics

- **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
Characteristics

- Physical type
  - Semiconductor (RAM)
  - Magnetic (Disk & Tape)
  - Optical (CD & DVD)

- Physical characteristics
  - Decay
  - Volatility
  - Erasable
  - Power consumption

- Organisation
  - Physical arrangement of bits into words
  - Not always obvious (e.g. interleaved)
Hierarchy List

- Registers
- L1 Cache
- L2 Cache
- L3 Cache
- Main memory
- Disk cache
- Disk
- Optical
- Tape
Cache Memory
Cache

- Small amount of fast memory
- Sits between normal main memory and CPU
- May be located on CPU chip or module
Cache and Main Memory

(a) Single cache

(b) Three-level cache organization
Cache/Main Memory Structure

(a) Cache

(b) Main memory

Line Number | Tag | Block
---|---|---
0 | | 
1 | | 
2 | | 
C – 1 | | 

Memory address

0 | 
1 | 
2 | 
3 | 

Block (K words)

Block

Word Length

$2^n - 1$
Cache operation – overview

- CPU requests contents of memory location
- Check cache for this data
  - If present:
    - get from cache
    - else read required block from main memory to cache
- Then deliver to CPU
- Cache includes tags to identify which block of main memory is in each cache slot
Cache Design

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

- Where does cache sit?
  - Between processor and virtual MMU
  - Between MMU and main memory

- Logical cache (virtual cache) stores data using virtual addresses
  - Processor accesses cache directly, not thorough physical cache
  - Cache access faster, before MMU address translation
  - Virtual addresses use same address space for different applications
    - Must flush cache on each context switch

- Physical cache stores data using main memory physical addresses
Size does matter

- **Cost**
  - More cache is expensive

- **Speed**
  - More cache is faster (up to a point)
  - Checking cache for data takes time
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 \( s-r \)

\[
\begin{array}{|c|c|c|}
\hline
\text{Tag} & \text{s-r} & \text{Line or Slot} \quad r & \text{Word} \quad w \\
\hline
8 & 14 & 2 \\
\hline
\end{array}
\]
Direct Mapping from Cache to Main Memory

First $m$ blocks of main memory (equal to size of cache)

(a) Direct mapping

$b = \text{length of block in bits}$
$t = \text{length of tag in bits}$
Direct Mapping Cache Organization

Memory Address
- Tag
- Line
- Word

Cache
- Tag
- Data

Main Memory
- W0
- W1
- W2
- W3

Compare
- 1 if match
- 0 if no match (hit in cache)
- 0 if match
- 1 if no match (miss in cache)
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
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

<table>
<thead>
<tr>
<th>Tag</th>
<th>22 bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>2 bit</td>
</tr>
</tbody>
</table>
Fully Associative Cache Organization

Memory Address

Tag
Word

Cache

Tag
Data

L_0

L_j

L_{m-1}

Main Memory

W_0
W_1
W_2
W_3

B_0

W_{4j}
W_{(4j+1)}
W_{(4j+2)}
W_{(4j+3)}

B_j

Compare

1 if match
0 if no match

(hit in cache)

0 if match
1 if no match

(miss in cache)
Associative Mapping Address Structure

- 22 bit tag stored with each 32 bit block of data
- Compare tag field with tag entry in cache to check for hit
- Least significant 2 bits of address identify which 16 bit word is required from 32 bit data block

- **e.g.**
  - **Address**  
  - **Tag**  
  - **Data**  
  - **Cache line**

<table>
<thead>
<tr>
<th>Address</th>
<th>Tag</th>
<th>Data</th>
<th>Cache line</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFFFFFC</td>
<td>FFFFFFFC</td>
<td>24682468</td>
<td>3FFF</td>
</tr>
</tbody>
</table>
Associative Mapping Summary

- Address length = \((s + w)\) bits
- Number of addressable units = \(2^{s+w}\) words or bytes
- Block size = line size = \(2^w\) words or bytes
- Number of blocks in main memory = \(2^{s+w}/2^w = 2^s\)
- Number of lines in cache = undetermined
- Size of tag = \(s\) bits
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 i
- e.g. 2 lines per set
  - 2 way associative mapping
  - A given block can be in one of 2 lines in only one set
Mapping From Main Memory to Cache: v Associative
Mapping From Main Memory to Cache: k-way Associative
K-Way Set Associative Cache Organization
Set Associative Mapping
Address Structure

- 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
  - 1FF 7FFC | 1FF | 12345678 | 1FFF
  - 001 7FFC | 001 | 11223344 | 1FFF
Set Associative Mapping Summary

- Address length = \((s + w)\) bits
- Number of addressable units = \(2^{s+w}\) words or bytes
- Block size = line size = \(2^w\) words or bytes
- Number of blocks in main memory = \(2^d\)
- Number of lines in set = \(k\)
- Number of sets = \(v = 2^d\)
- Number of lines in cache = \(kv = k \times 2^d\)
- Size of tag = \((s - d)\) bits
Replacement Algorithms

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

- Associative & Set Associative
  - Hardware implemented algorithm (speed)
  - Least Recently used (LRU)
    - but in 2 way set associative “Which of the 2 block is LRU?”
  - First in first out (FIFO)
  - Least frequently used
    - replace block which has had fewest hits
  - Random
Write Policy

- **Not overwrite** a cache block if main memory is up to date
  - I/O may address main memory directly

- **Write through**
  - All writes go to main memory as well as cache
  - Lots of traffic
  - Slows down writes

- **Write back**
  - Updates initially made in cache only and *update bit* is set
  - 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
Line Size

- Retrieve not only desired word but a number of adjacent words as well
- Increased block size will increase hit ratio at first
  - the principle of locality
- Hit ratio will decreases as block becomes even bigger
  - Probability of using newly fetched information becomes less than probability of reusing replaced
- Larger blocks
  - Reduce number of blocks that fit in cache
  - Data overwritten shortly after being fetched
  - Each additional word is less local so less likely to be needed
- No definitive optimum value has been found
- 8 to 64 bytes seems reasonable
- For HPC systems, 64 and 128 byte most common
Multilevel Caches

- High logic density enables caches on chip
  - Faster than bus access
  - Frees bus for other transfers
- Common to use both on and off chip cache
  - L1 on chip, L2 off chip in static RAM
  - L2 access much faster than DRAM or ROM
  - L2 often uses separate data path
  - L2 may now be on chip
  - Resulting in L3 cache
    - Bus access or now on chip...
Unified v Split Caches

- One cache for data and instructions or two, one for data and one for instructions

- Advantages of unified cache
  - Higher hit rate
    - Balances load of instruction and data fetch
    - Only one cache to design & implement

- Advantages of split cache
  - Eliminates cache contention between instruction fetch/decode unit and execution unit
    - Important in pipelining
Internal Memory
## Semiconductor Memory Types

<table>
<thead>
<tr>
<th>Memory Type</th>
<th>Category</th>
<th>Erasure</th>
<th>Write Mechanism</th>
<th>Volatility</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random-access memory (RAM)</td>
<td>Read-write</td>
<td>Electrically, byte-level</td>
<td>Electrically</td>
<td>Volatile</td>
</tr>
<tr>
<td>Read-only memory (ROM)</td>
<td>Read-only</td>
<td>Not possible</td>
<td>Masks</td>
<td></td>
</tr>
<tr>
<td>Programmable ROM (PROM)</td>
<td>Read-only memory</td>
<td>Not possible</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Erasable PROM (EPROM)</td>
<td>Read-mostly memory</td>
<td>UV light, chip-level</td>
<td>Electrically</td>
<td>Nonvolatile</td>
</tr>
<tr>
<td>Electrically Erasable PROM (EEPROM)</td>
<td>Read-mostly memory</td>
<td>Electrically, byte-level</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flash memory</td>
<td></td>
<td>Electrically, block-level</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Semiconductor Memory

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

(a) Write

(b) Read
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
- Essentially analog
  - Level of charge determines value
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
- Digital
  - Uses flip-flops
SRAM v DRAM

- Both volatile
  - Power needed to preserve data
- Dynamic cell
  - Simpler to build, smaller
  - More dense
  - Less expensive
  - Needs refresh
  - Larger memory units
- Static
  - Faster
  - Cache
Read Only Memory (ROM)

- Permanent storage
  - Nonvolatile
- Microprogramming
- 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

- 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)
256kByte Module Organisation
1MByte Module Organisation

Memory Address Register (MAR)

Chip Group Enable
Select 1 of 4 Groups

Memory Buffer Register (MBR)

Bit 1
All chips 512 words by 512 bits, 2-terminal cells

Bit 8

Groups A B C D

A1 1/512 1/512
B1 1/512
A2
B2
A7 1/512
B7
C1
C8
D1
D8

1 2 7 8
Interleaved Memory

- Main memory is composed of a collection of DRAM memory chips that can be grouped together to form a memory bank.
- It is possible to organize the memory banks in a way known as interleaved memory.
- Each bank is independently able to service a memory read or write request, so that a system with K banks can service K requests simultaneously, increasing memory read or write rates by a factor of K.
- If consecutive words of memory are stored in different banks, then the transfer of a block of memory is speeded up.
Advanced DRAM Organization

- Basic DRAM same since first RAM chips
- Enhanced DRAM
  - Contains small SRAM as well
  - SRAM holds last line read (c.f. Cache!)
- Cache DRAM
  - Larger SRAM component
  - Use as cache or serial buffer
Synchronous DRAM (SDRAM)

- 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
- DDR-SDRAM sends data twice per clock cycle (leading & trailing edge)
SDRAM

[Diagram of SDRAM architecture with labels for CKE, CLK, Address buffers, Control decoder, Cell array memory banks, Data I/O buffers, etc.]
**DDR SDRAM**

- SDRAM can only send data once per clock
- Double-data-rate SDRAM can send data twice per clock cycle
  - Rising edge and falling edge
External Memory
Types of External Memory

- Magnetic Disk
  - RAID
  - Removable
- Optical
  - CD-ROM
  - CD-Recordable (CD-R)
  - CD-R/W
  - DVD
- Magnetic Tape
Magnetic Disk

- Disk substrate coated with magnetizable material
- Substrate used to be aluminium
- Now glass
  - Improved surface uniformity
    - Increases reliability
  - Reduction in surface defects
    - Reduced read/write errors
  - Lower flight heights (See later)
  - Better stiffness
  - Better shock/damage resistance
Read and Write Mechanisms

- Recording & retrieval via conductive coil called a head
- May be single read/write head or separate ones
- During read/write, head is stationary, platter rotates

Write
- Current through coil produces magnetic field
- Pulses sent to head
- Magnetic pattern recorded on surface below
  - Higher storage density and speed
Read and Write Mechanisms

- **Read (traditional)**
  - Magnetic field moving relative to coil produces current
  - Coil is the same for read and write

- **Read (contemporary)**
  - Separate read head, close to write head
  - Partially shielded magneto resistive (MR) sensor
  - Electrical resistance depends on direction of magnetic field
  - High frequency operation
    - Higher storage density and speed
Data Organization and Formatting

- Concentric rings or tracks
  - Gaps between tracks
  - Reduce gap to increase capacity
  - Same number of bits per track (variable packing density)
  - Constant angular velocity
- Tracks divided into sectors
  - Minimum block size is one sector
  - May have more than one sector per block
Disk Velocity

- Bit near centre of rotating disk passes fixed point slower than bit on outside of disk
- Increase spacing between bits in different tracks
- Rotate disk at constant angular velocity (CAV)
  - Gives pie shaped sectors and concentric tracks
  - Individual tracks and sectors addressable
  - Move head to given track and wait for given sector
  - Waste of space on outer tracks
    - Lower data density
- Can use zones to increase capacity
  - Each zone has fixed bits per track
  - More complex circuitry
Disk Layout Methods Diagram

(a) Constant angular velocity

(b) Multiple zoned recording
Finding Sectors

- Must be able to identify start of track and sector
- Format disk
  - Additional information not available to user
  - Marks tracks and sectors
Characteristics

- Fixed (rare) or movable head
- Removable or fixed
- Single or double (usually) sided
- Single or multiple platter
- Head mechanism
  - Contact (Floppy)
  - Fixed gap
  - Flying (Winchester)
Fixed/Movable Head Disk

- Fixed head
  - One read write head per track
  - Heads mounted on fixed ridged arm

- Movable head
  - One read write head per side
  - Mounted on a movable arm
Removable or Not

- Removable disk
  - Can be removed from drive and replaced with another disk
  - Provides unlimited storage capacity
  - Easy data transfer between systems
- Nonremovable disk
  - Permanently mounted in the drive
Multiple Platter

- One head per side
- Heads are joined and aligned
- Aligned tracks on each platter form cylinders
- Data is striped by cylinder
  - reduces head movement
  - Increases speed (transfer rate)
Winchester Hard Disk

- Developed by IBM in Winchester (USA)
- Sealed unit
- One or more platters (disks)
- Heads fly on boundary layer of air as disk spins
- Very small head to disk gap
- Getting more robust
- Universal
- Cheap
- Fastest external storage
- Getting larger all the time
  - 250 Gigabyte now easily available
Speed

- Seek time
  - Moving head to correct track
- (Rotational) latency
  - Waiting for data to rotate under head
- Access time = Seek + Latency
- Transfer rate
RAID

- Redundant Array of Independent Disks
- Redundant Array of Inexpensive Disks
- 6 levels in common use
- Not a hierarchy
- Set of physical disks viewed as single logical drive by O/S
- Data distributed across physical drives
- Can use redundant capacity to store parity information
### RAID 0

- No redundancy
- Data striped across all disks
- Round Robin striping
- Increase speed
  - Multiple data requests probably not on same disk
  - Disks seek in parallel
  - A set of data is likely to be striped across multiple disks

### RAID 1

- Mirrored Disks
- Data is striped across disks
- 2 copies of each stripe on separate disks
- Read from either
- Write to both
- Recovery is simple
  - Swap faulty disk & re-mirror
  - No down time
- Expensive
RAID

RAID 2
- Disks are synchronized
- Very small stripes
  - Often single byte/word
- Error correction calculated across corresponding bits on disks
- Multiple parity disks store Hamming code error
- Lots of redundancy (expensive)

RAID 3
- Similar to RAID 2
- Only one redundant disk, no matter how large the array
- Simple parity bit for each set of corresponding bits
- Data on failed drive can be reconstructed from surviving data and parity info
- Very high transfer rates
### RAID

#### RAID 4
- Each disk operates independently
- Good for high I/O request rate
- Large stripes
- Bit by bit parity calculated across stripes on each disk
- Parity stored on parity disk

#### RAID 5
- Like RAID 4
- Parity striped across all disks
- Round robin allocation for parity stripe
- Avoids RAID 4 bottleneck at parity disk

#### RAID 6
- Two parity calculations
- Stored in separate blocks on different disks
- User requirement of N disks needs N+2
- High data availability
- Three disks need to fail for data loss
- Significant write penalty
- Commonly used in network servers
RAID 0, 1, 2

(a) RAID 0 (non-redundant)

(b) RAID 1 (mirrored)

(c) RAID 2 (redundancy through Hamming code)
RAID 3 & 4

(d) RAID 3 (bit-interleaved parity)

(e) RAID 4 (block-level parity)
Optical Storage CD-ROM

- Originally for audio
- 650Mbytes giving over 70 minutes audio
- Polycarbonate coated with highly reflective coat, usually aluminium
- Data stored as pits
- Read by reflecting laser
- Constant packing density
- Constant linear velocity
CD Operation

![Diagram](image)

- Protective acrylic
- Label
- Polycarbonate plastic
- Land
- Pit
- Aluminum
- Laser transmit/receive
CD-ROM Drive Speeds

- Audio is single speed
  - Constant linear velocity
  - 1.2 ms$^{-1}$
  - Track (spiral) is 5.27km long
  - Gives 4391 seconds = 73.2 minutes

- Other speeds are quoted as multiples
- e.g. 24x
- Quoted figure is maximum drive can achieve
CD-ROM Format

- Mode 0 = blank data field
- Mode 1 = 2048 byte data + error correction
- Mode 2 = 2336 byte data
Random Access on CD-ROM

- Difficult
- Move head to rough position
- Set correct speed
- Read address
- Adjust to required location
- (Yawn!)
CD-ROM for & against

- Large capacity (?)
- Easy to mass produce
- Removable
- Robust

- Expensive for small runs
- Slow
- Read only
Other Optical Storage

- CD-Recordable (CD-R)
  - WORM
  - Now affordable
  - Compatible with CD-ROM drives
- CD-RW
  - Erasable
  - Getting cheaper
  - Mostly CD-ROM drive compatible
  - Phase change
    - Material has two different reflectivities in different phase states
DVD - what’s in a name?

- Digital Video Disk
  - Used to indicate a player for movies
    - Only plays video disks
- Digital Versatile Disk
  - Used to indicate a computer drive
    - Will read computer disks and play video disks
DVD - technology

- Multi-layer
- Very high capacity (4.7G per layer)
- Full length movie on single disk
  - Using MPEG compression
- Finally standardized
- Movies carry regional coding
- Players only play correct region films
- Can be “fixed”
High Definition Optical Disks

- Designed for high definition videos
- Much higher capacity than DVD
  - Shorter wavelength laser
    - Blue-violet range
  - Smaller pits
- HD-DVD
  - 15GB single side single layer
- Blue-ray
  - Data layer closer to laser
    - Tighter focus, less distortion, smaller pits
  - 25GB on single layer
  - Available read only (BD-ROM), Recordable once (BR-R) and re-recordable (BR-RE)
Optical Memory Characteristics

- CD: Beam spot diameter 2.11 μm
- DVD: Beam spot diameter 1.32 μm
- Blu-ray: Laser wavelength 780 nm, pit depth 0.58 μm, laser wavelength 405 nm
- Data layer thickness: 1.2 μm
- DVD: Laser wavelength 650 nm, pit depth 0.6 μm

100
Magnetic Tape

- Serial access
- Slow
- Very cheap
- Backup and archive
- Linear Tape-Open (LTO) Tape Drives
  - Developed late 1990s
  - Open source alternative to proprietary tape systems
Input Output
Input/Output Problems

- Wide variety of peripherals
  - Delivering different amounts of data
  - At different speeds
  - In different formats
- All slower than CPU and RAM
- Need I/O modules

Input/Output module
- Interface to CPU and Memory
- Interface to one or more peripherals
Generic Model of I/O Module
External Devices

- Human readable
  - Screen, printer, keyboard
- Machine readable
  - Monitoring and control
- Communication
  - Modem
  - Network Interface Card (NIC)
External Device Block Diagram

Control signals from I/O module

Status signals to I/O module

Data bits to and from I/O module

Control Logic

Buffer

Transducer

Data (device-unique) to and from environment
I/O Module Function

- Control & Timing
- CPU Communication
- Device Communication
- Data Buffering
- Error Detection
I/O Steps

- CPU checks I/O module device status
- I/O module returns status
- If ready, CPU requests data transfer
- I/O module gets data from device
- I/O module transfers data to CPU
- Variations for output, DMA, etc.
I/O Module Diagram

- Interface to System Bus
- Interface to External Device

- Data Registers
- Status/Control Registers
- External Device Interface Logic
- Control Lines
- Address Lines
- Data Lines

- External Device Interface Logic
- Data
- Status
- Control
I/O Module Decisions

- Hide or reveal device properties to CPU
- Support multiple or single device
- Control device functions or leave for CPU
- Also O/S decisions
  - e.g. Unix treats everything it can as a file
Input Output Techniques

- Programmed
- Interrupt driven
- Direct Memory Access (DMA)
Three Techniques for Input of a Block of Data

(a) Programmed I/O

Issue Read command to I/O module

Read status of I/O module

I/O → CPU

Check status

CPU → I/O

Do something else

Read word from I/O Module

I/O → CPU

Write word into memory

CPU → memory

Next instruction

(b) Interrupt-driven I/O

Issue Read command to I/O module

Read status of I/O module

I/O → CPU

Check status

CPU → DMA

Do something else

Read word from I/O Module

I/O → CPU

Write word into memory

CPU → memory

Next instruction

(c) Direct memory access

Issue Read block command to I/O module

Read status of DMA module

DMA → CPU

Error condition

Ready

Interrupt

Error condition

Next instruction
Programmed I/O

- CPU has direct control over I/O
  - Sensing status
  - Read/write commands
  - Transferring data
- CPU waits for I/O module to complete operation
- Wastes CPU time
Programmed I/O - detail

- CPU requests I/O operation
- I/O module performs operation
- I/O module sets status bits
- CPU checks status bits periodically
- I/O module does not inform CPU directly
- I/O module does not interrupt CPU
- CPU may wait or come back later
I/O Commands

- CPU issues address
  - Identifies module (& device if >1 per module)
- CPU issues command
  - Control - telling module what to do
    - e.g. spin up disk
  - Test - check status
    - e.g. power? Error?
- Read/Write
  - Module transfers data via buffer from/to device
Addressing I/O Devices

- Under programmed I/O data transfer is very like memory access (CPU viewpoint)
- Each device given unique identifier
- CPU commands contain identifier (address)
I/O Mapping

- Memory mapped I/O
  - Devices and memory share an address space
  - I/O looks just like memory read/write
  - No special commands for I/O
    - Large selection of memory access commands available

- Isolated I/O
  - Separate address spaces
  - Need I/O or memory select lines
  - Special commands for I/O
    - Limited set
Interrupt Driven I/O

- Overcomes CPU waiting
- No repeated CPU checking of device
- I/O module interrupts when ready
Interrupt Driven I/O - Basic Operation

- CPU issues read command
- I/O module gets data from peripheral whilst CPU does other work
- I/O module interrupts CPU
- CPU requests data
- I/O module transfers data
Simple Interrupt Processing

Hardware

1. Device controller or other system hardware issues an interrupt
2. Processor finishes execution of current instruction
3. Processor signals acknowledgment of interrupt
4. Processor pushes PSW and PC onto control stack
5. Processor loads new PC value based on interrupt

Software

1. Save remainder of process state information
2. Process interrupt
3. Restore process state information
4. Restore old PSW and PC
CPU Viewpoint

- Issue read command
- Do other work
- Check for interrupt at end of each instruction cycle
- If interrupted:-
  - Save context (registers)
  - Process interrupt
    - Fetch data & store
- See Operating Systems notes
Changes in Memory and Registers for an Interrupt

(a) Interrupt occurs after instruction at location N

(b) Return from interrupt
Design Issues

- How do you identify the module issuing the interrupt?
- How do you deal with multiple interrupts?
  - i.e. an interrupt handler being interrupted
Identifying Interrupting Module

- Different line for each module
  - PC
  - Limits number of devices
- Software poll
  - CPU asks each module in turn
  - Slow
Identifying Interrupting Module

- **Daisy Chain or Hardware poll**
  - Interrupt Acknowledge sent down a chain
  - Module responsible places vector on bus
  - CPU uses vector to identify handler routine

- **Bus Master**
  - Module must claim the bus before it can raise interrupt
  - e.g. PCI & SCSI
Multiple Interrupts

- Each interrupt line has a priority
- Higher priority lines can interrupt lower priority lines
- If bus mastering only current master can interrupt
Direct Memory Access

- Interrupt driven and programmed I/O require active CPU intervention
  - Transfer rate is limited
  - CPU is tied up
- DMA is the answer

- Additional Module (hardware) on bus
- DMA controller takes over from CPU for I/O
Typical DMA Module Diagram
DMA Operation

- CPU tells DMA controller:
  - Read/Write
  - Device address
  - Starting address of memory block for data
  - Amount of data to be transferred
- CPU carries on with other work
- DMA controller deals with transfer
- DMA controller sends interrupt when finished
DMA Transfer - Cycle Stealing

- DMA controller takes over bus for a cycle
- Transfer of one word of data
- Not an interrupt
  - CPU does not switch context
- CPU suspended just before it accesses bus
  - i.e. before an operand or data fetch or a data write
- Slows down CPU but not as much as CPU doing transfer
DMA and Interrupt Breakpoints During an Instruction Cycle
DMA Configurations (1)

- Single Bus, Detached DMA controller
- Each transfer uses bus twice
  - I/O to DMA then DMA to memory
- CPU is suspended twice
DMA Configurations (2)

- Single Bus, Integrated DMA controller
- Controller may support >1 device
- Each transfer uses bus once
  - DMA to memory
- CPU is suspended once
DMA Configurations (3)

- Separate I/O Bus
- Bus supports all DMA enabled devices
- Each transfer uses bus once
  - DMA to memory
- CPU is suspended once
I/O Channels

- I/O devices getting more sophisticated
- e.g. 3D graphics cards
- CPU instructs I/O controller to do transfer
- I/O controller does entire transfer
- Improves speed
  - Takes load off CPU
  - Dedicated processor is faster
I/O Channel Architecture

(a) Selector
(b) Multiplexor
Interfacing

- Connecting devices together
- Bit of wire?
- Dedicated processor/memory/buses?
- E.g. FireWire, InfiniBand
IEEE 1394 FireWire

- High performance serial bus
- Fast
- Low cost
- Easy to implement
- Also being used in digital cameras, VCRs and TV
FireWire Configuration

- Daisy chain
- Up to 63 devices on single port
  - Really 64 of which one is the interface itself
- Up to 1022 buses can be connected with bridges
- Automatic configuration
- No bus terminators
- May be tree structure
Simple FireWire Configuration

Diagram showing a simple FireWire configuration with connected devices:
- Stereo interface
- CPU
- Magnetic disk
- CD-ROM
- Digital camera
- Scanner
- Printer
InfiniBand

- I/O specification aimed at high end servers
  - Merger of Future I/O (Cisco, HP, Compaq, IBM) and Next Generation I/O (Intel)
- Version 1 released early 2001
- Architecture and spec. for data flow between processor and intelligent I/O devices
- Intended to replace PCI in servers
- Increased capacity, expandability, flexibility
InfiniBand Architecture

- Remote storage, networking and connection between servers
- Attach servers, remote storage, network devices to central fabric of switches and links
- Greater server density
- Scalable data centre
- Independent nodes added as required
- I/O distance from server up to
  - 17m using copper
  - 300m multimode fibre optic
  - 10km single mode fibre
- Up to 30Gbps
InfiniBand Switch Fabric

IB = InfiniBand
HCA = host channel adapter
TCA = target channel adapter
InfiniBand Operation

- 16 logical channels (virtual lanes) per physical link
- One lane for management, rest for data
- Data in stream of packets
- Virtual lane dedicated temporarily to end to end transfer
- Switch maps traffic from incoming to outgoing lane