Computer Architecture

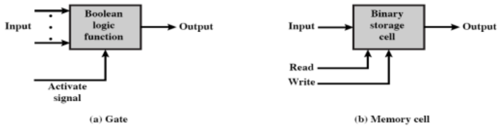
Prof. Dr. Nizamettin AYDIN

naydin@yildiz.edu.tr

http://www.yildiz.edu.tr/~naydin

Fundamental Computer Elements

Fundamental computer elements

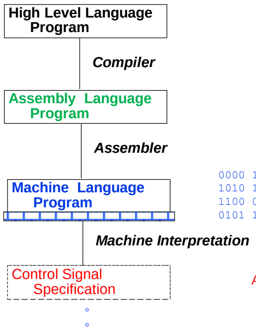


(a) Gate

(b) Memory cell

- Data processing
  - Provided by gates
- Data storage
  - Provided by memory cells
- Data movement
  - The paths between components are used to move data from/to memory
- Control
  - The paths between components can carry control signals

Levels of Representation


High Level Language Program
Compiler
Assembly Language Program
Assembler
Machine Language Program
Machine Interpretation
Control Signal Specification

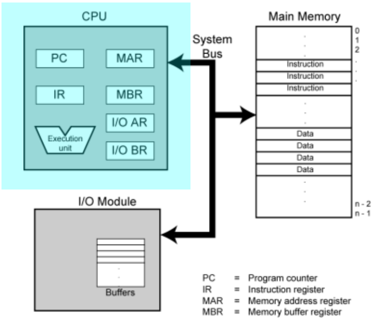
```
temp = v[k];
v[k] = v[k+1];
v[k+1] = temp;

lw      $15,    0($2)
lw      $16,    4($2)
sw      $16,    0($2)
sw      $15,    4($2)

0000 1001 1100 0110 1010 1111 0101 1000
1010 1111 0101 1000 0000 1001 1100 0110
1100 0110 1010 1111 0101 1000 0000 1001
0101 1000 0000 1001 1100 0110 1010 1111

ALUOP[0:3] <= InstReg[9:11] & MASK
```

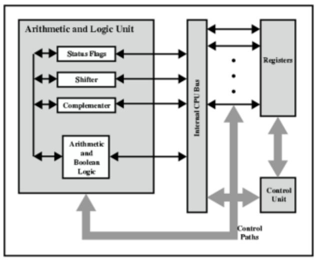
Computer Components-CPU



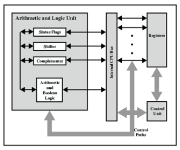
PC = Program counter
IR = Instruction register
MAR = Memory address register
MBR = Memory buffer register
I/O AR = Input/output address register
I/O BR = Input/output buffer register

CPU Structure

- CPU must:
  - Fetch instructions
  - Interpret instructions
  - Fetch data
  - Process data
  - Write data



Registers



- CPU must have some working space (temporary storage)
  - Called registers
- Number and function vary between processor designs
- One of the major design decisions
- Top level of memory hierarchy

Registers in the µP perform two roles:

- User-visible registers
  - Enable the machine- or assembly language programmer to minimize main memory references by optimizing use of registers
- Control and status registers
  - Used by the control unit to control the operation of the processor and by priviliged, operating system programs to control the execution of programs

User Visible Registers

- General Purpose registers
- Data registers
- Address registers
- Condition Codes (flags)

Condition Code Registers

- Sets of individual bits
  - e.g. result of last operation was zero
- Can be read (implicitly) by programs
  - e.g. Jump if zero
- Can not (usually) be set by programs

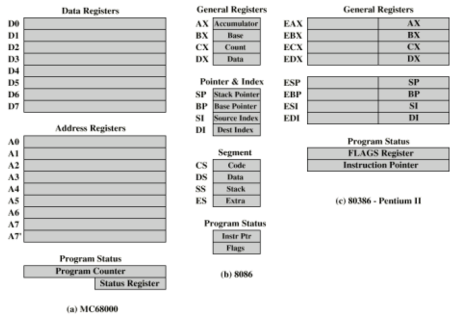
Control & Status Registers

- Program Counter (PC)
  - Contains the address of an instruction to be fetched
- Instruction Decoding Register (IR)
  - Contains the instruction most recently fetched
- Memory Address Register (MAR)
  - Contains the addres of location in memory
- Memory Buffer Register (MBR)
  - Contains a word or data to be written to memory or the word most recently read

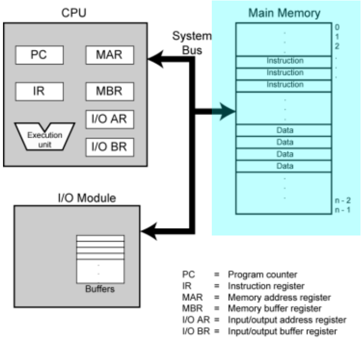
Program Status Word

- A set of bits containing status information
- Includes Condition Codes (flags)
  - Sign
    - sign of last result
  - Zero
    - set when the result is 0
  - Carry
    - set if an operation resulted in a carry (addition) into or borrow (subtraction) out of a high order bit
  - Equal
    - set if a logical compare result is equality
  - Overflow
    - used to indicate arithmetic overflow
  - Interrupt enable/disable
    - used to enable or disable interrupts

Example Register Organizations



Computer Components-Memory

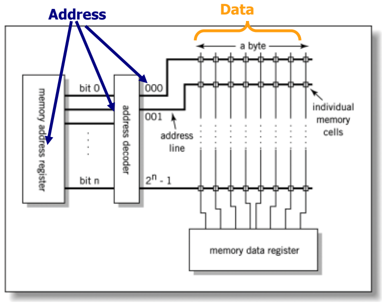


PC = Program counter
IR = Instruction register
MAR = Memory address register
MBR = Memory buffer register
I/O AR = Input/output address register
I/O BR = Input/output buffer register

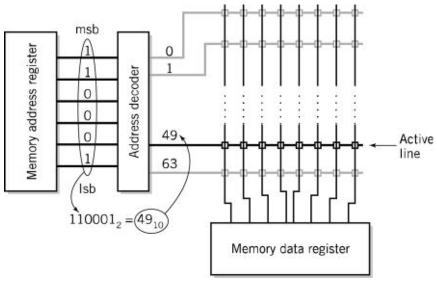
Operation of Memory

- Each memory location has a unique address
- Address from an instruction is copied to the MAR which finds the location in memory
- CPU determines if it is a store or retrieval
- Transfer takes place between the MBR and memory
  - MBR is a two way register

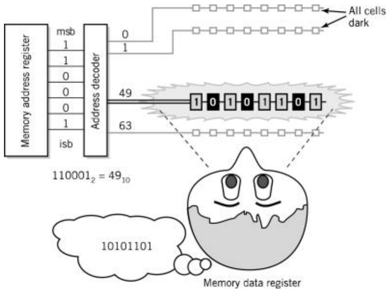
Relationship between MAR, MBR and Memory



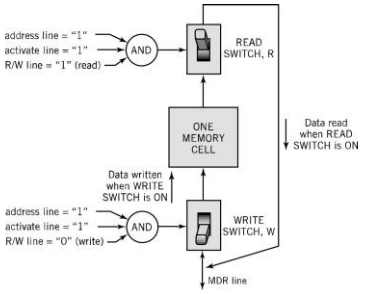
MAR-MBR Example



Visual Analogy of Memory



Individual Memory Cell



Memory Capacity

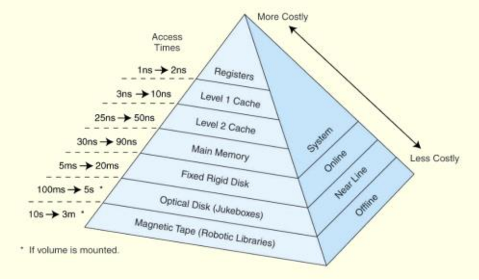
- Determined by two factors
  - Number of bits in the MAR
    - 2^K where K = width of the register in bits
  - Size of the address portion of the instruction
    - 4 bits allows 16 locations
    - 8 bits allows 256 locations
    - 32 bits allows 4,294,967,296 or 4 GB
- Important for performance
  - Insufficient memory can cause a processor to work at 50% below performance

Memory Hierarchy

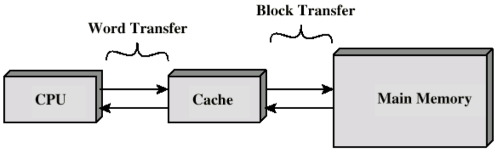
- Registers
  - In CPU
- Internal or Main memory
  - May include one or more levels of cache
  - "RAM"
- External memory
  - Backing store

Memory Hierarchy

- This storage organization can be thought of as a pyramid:



Cache



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

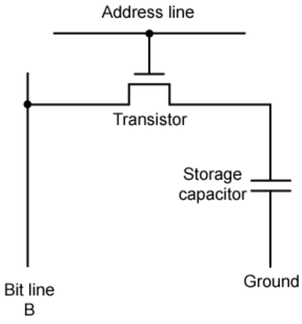
Virtual Memory

- Cache memory enhances performance by providing faster memory access speed.
- Virtual memory enhances performance by providing greater memory capacity, without the expense of adding main memory.
- Instead, a portion of a disk drive serves as an extension of main memory.
- If a system uses paging, virtual memory partitions main memory into individually managed *page frames*, that are written *(or paged)* to disk when they are not immediately needed.

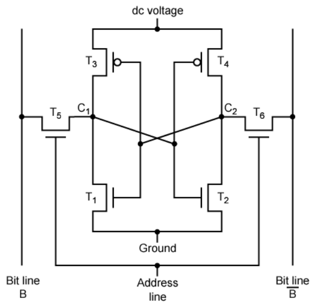
RAM: Random Access Memory

- DRAM (Dynamic RAM)
  - Most common, cheap
  - Volatile: must be refreshed (recharged with power) 1000’s of times each second
- SRAM (static RAM)
  - Faster than DRAM and more expensive than DRAM
  - Volatile
  - Frequently small amount used in *cache memory* for high-speed access used

Dynamic RAM Structure



Stating RAM Structure



Read Only Memory (ROM)

- Permanent storage
  - Nonvolatile
- Used in...
  - Microprogramming
  - Library subroutines
  - Systems programs (BIOS)
  - Function tables

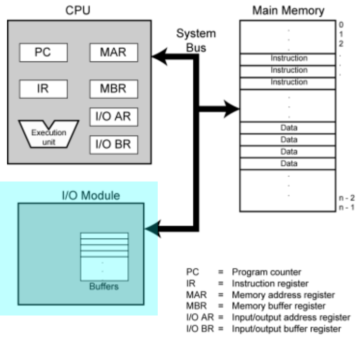
Types of ROM

- 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

Types of External Memory

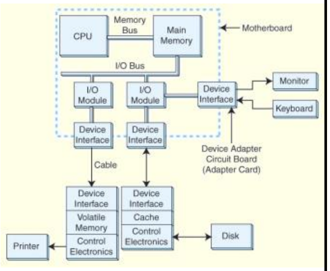
- SSD
  - Fast
  - Expensive (relatively)
- Magnetic Disk
  - RAID
  - Removable
- Optical
  - CD-ROM
  - CD-Recordable (CD-R)
  - CD-R/W
  - DVD
- Magnetic Tape

Computer Components-I/O



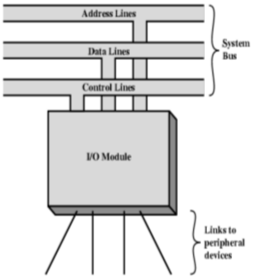
Input/Output Problems

- 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

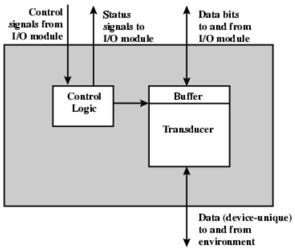
I/O Module Block Diagram



- Interface to CPU and Memory
- Interface to one or more peripherals
- I/O Module Function:
  - Control & Timing
  - CPU Communication
  - Device Communication
  - Data Buffering
  - Error Detection

External Devices

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

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

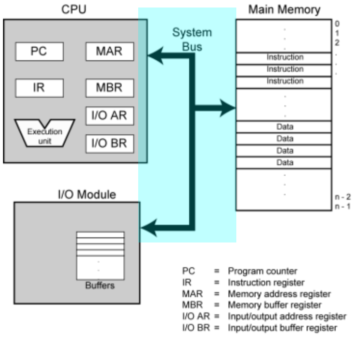
I/O Steps

- 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 Architectures

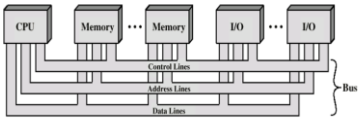
- I/O can be controlled in four general ways:
  - Programmed I/O
    - Reserves a register for each I/O device.
    - Each register is continually polled to detect data arrival.
  - Interrupt-Driven I/O
    - Allows the CPU to do other things until I/O is requested.
  - Direct Memory Access (DMA)
    - Offloads I/O processing to a special-purpose chip that takes care of the details.
  - Channel I/O
    - Uses dedicated I/O processors.

Computer Components- Bus



Bus

- The physical connection that makes it possible to transfer data from one location in the computer system to another
- Group of electrical conductors for carrying signals from one location to another
- 4 kinds of signals
  - Data
    - Alphanumeric
    - Numerical
    - instructions
  - Addresses
  - Control signals
  - Power (sometimes)



Bus

- Connects CPU and Memory
- I/O peripherals: on the same bus as CPU/memory or separate bus
- Physical packaging commonly called *backplane*
  - Also called *system bus* or *external bus*
  - Example of *broadcast bus* (address bus)
  - Part of printed circuit board called *motherboard* that holds CPU and related components

Bus Characteristics

- Protocol
  - Documented agreement for communication
  - Specification that spells out the meaning of each line and each signal on each line
- Throughput, i.e., data transfer rate in bits per second
- Data width in bits carried simultaneously

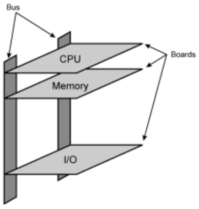
Bus types

- Data Bus
  - Carries data
  - Width is a key determinant of performance
    - 8, 16, 32, 64 bit
- Address bus
  - Identify the source or destination of data
  - Bus width determines maximum memory capacity of system
    - e.g. 8080 has 16 bit address bus giving 64k address space
- Control Bus
  - Control and timing information
    - Memory read/write; I/O read/write; Transfer acknowledge; Bus request; Bus grant; Interrupt request; Interrupt acknowledge; Clock; Reset

What do buses look like?

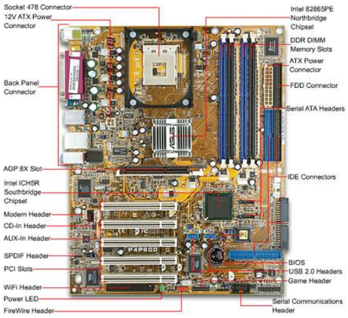
- Parallel lines on circuit boards
- Ribbon cables
- Strip connectors on mother boards
  - e.g. PCI
- Sets of wires

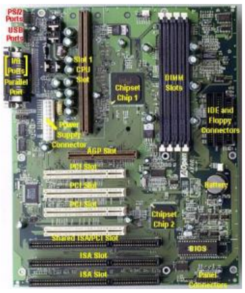
Physical Realization of
Bus Architecture

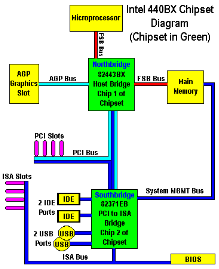


Motherboard

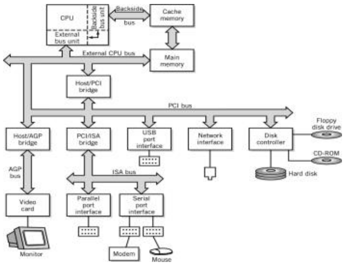
- Printed circuit board that holds CPU and related components including backplane






Microprocessor
Intel 440BX Chipset Diagram (Chipset in Green)
FSB Bus
AGP Graphics Slot
AGP Bus
Northbridge 82443BX Host Bridge Chip 1 of Chipset
FSB Bus
Main Memory
PCI Slots
PCI Bus
ISA Slots
System MGMT Bus
2 IDE Ports
IDE
IDE
2 USB Ports
USB
USB
Southbridge 82371EB PCI to ISA Bridge Chip 2 of Chipset
ISA Bus
BIOS

Typical PC Interconnections



Bus interface bridges connect different bus types

Instructions

- Instruction:
  - Direction given to a computer
  - Causes electrical signals to be sent through specific circuits for processing
- The operation of the processor is determined by the instructions it executes,
  - which is referrred as machine instructions or computer instructions
- The collection of different instructions that the processor execute is referred as the processor’s instruction set.

What is an Instruction Set?

- The complete collection of instructions that are understood by a CPU
  - Machine Code
  - Binary
- Usually represented by assembly codes
- Differentiates computer architecture by the
  - Number of instructions
  - Complexity of operations performed by individual instructions
  - Data types supported
  - Format (layout, fixed vs. variable length)
  - Use of registers
  - Addressing (size, modes)

Instruction Elements

OPCODE	Source OPERAND	Result OPERAND

- Operation code (OPCODE): task
  - Do this
- Source OPERAND reference(s)
  - To this
- Result OPERAND reference
  - Put the answer here

Addresses

- Next Instruction Reference
  - When you have done that, do this...

Instruction Elements

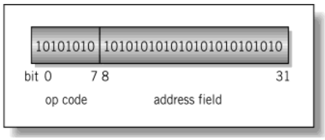
- Source and Result Operands can be in one of the following areas:
  - Main memory
  - Virtual memory
  - Cache
  - CPU register
  - I/O device

49

Instruction Format

- *Machine-specific* template that specifies
  - Length of the op code
  - Number of operands
  - Length of operands

Simple 32-bit Instruction Format



50

Instruction Types

- Data Transfer (load, store)
  - Most common, greatest flexibility
  - Involve memory and registers
  - What's a *word* ? 16? 32? 64 bits?
- Arithmetic
  - Operators **+ - / * ^**
  - Integers and floating point
- Logical or Boolean
  - Relational operators: **> < =**
  - Boolean operators **AND, OR, XOR, NOR,** and **NOT**
- Single operand manipulation instructions
  - Negating, decrementing, incrementing

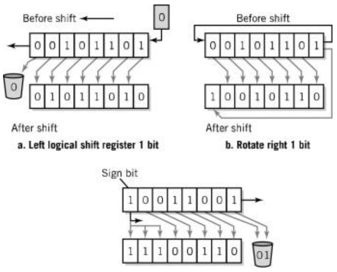
51

More Instruction Types

- Bit manipulation instructions
  - Flags to test for conditions
- Shift and rotate
- Program control
- Stack instructions
- Multiple data instructions
- I/O and machine control

52

Register Shifts and Rotates



a. Left logical shift register 1 bit

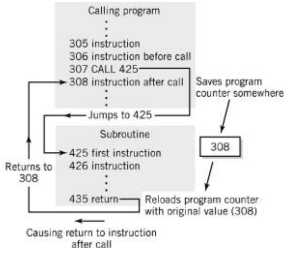
b. Rotate right 1 bit

c. Right arithmetic shift 2 bits

53

Program Control Instructions

- Program control
  - Jump and branch
  - Subroutine call and return

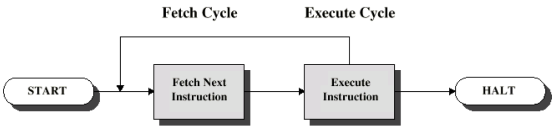


Calling program
305 instruction
306 instruction before call
307 CALL 425
308 instruction after call
Saves program counter somewhere
Jumps to 425
Subroutine
425 first instruction
426 instruction
308
Returns to 308
435 return
Reloads program counter with original value (308)
Causing return to instruction after call

54

Instruction Cycle

- Two steps:
  - Fetch
  - Execute



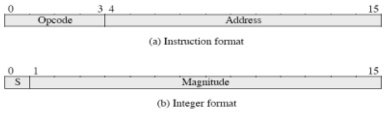
Fetch Cycle

- Program Counter (PC) holds address of next instruction to fetch
- Processor fetches instruction from memory location pointed to by PC
- Increment PC
  - Unless told otherwise
- Instruction loaded into Instruction Register (IR)
- Processor interprets instruction and performs required actions

Execute Cycle

- Processor-memory
  - data transfer between CPU and main memory
- Processor I/O
  - Data transfer between CPU and I/O module
- Data processing
  - Some arithmetic or logical operation on data
- Control
  - Alteration of sequence of operations
  - e.g. jump
- Combination of above

A simple example – A hypotetical machine



(a) Instruction format

(b) Integer format

Program Counter (PC) = Address of instruction
Instruction Register (IR) = Instruction being executed
Accumulator (AC) = Temporary storage

(c) Internal CPU registers

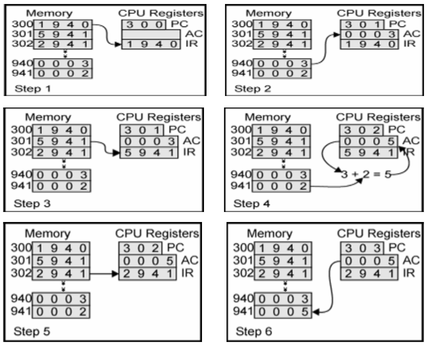
0001 = Load AC from Memory
0010 = Store AC to Memory
0101 = Add to AC from Memory

(d) Partial list of opcodes

A simple example –

- Next figure illustrates a partial program execution.
- It adds the contents of the memory word at address 940 to the contents of the memory word at address 941 and stores the result in the address 941.
- Here 3 instructions (3 fetch and 3 execute cycles) are required

Example of Program Execution


Memory
CPU Registers
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2
3 0 0 PC
AC
1 9 4 0 IR
Step 1
Memory
CPU Registers
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2
3 0 1 PC
0 0 0 3 AC
1 9 4 0 IR
Step 2
Memory
CPU Registers
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2
3 0 1 PC
0 0 0 3 AC
5 9 4 1 IR
Step 3
Memory
CPU Registers
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2
3 0 2 PC
0 0 0 5 AC
5 9 4 1 IR
3 + 2 = 5
Step 4
Memory
CPU Registers
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 2
3 0 2 PC
0 0 0 5 AC
2 9 4 1 IR
Step 5
Memory
CPU Registers
300 1 9 4 0
301 5 9 4 1
302 2 9 4 1
940 0 0 0 3
941 0 0 0 5
3 0 3 PC
0 0 0 5 AC
2 9 4 1 IR
Step 6