SPRING 2010

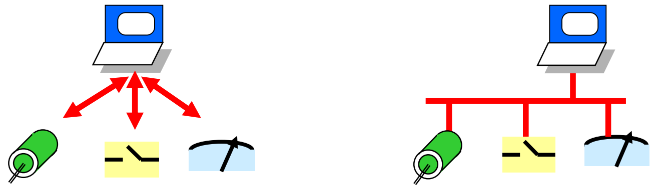
即時控制系統設計
Design of Real-Time Control Systems

Lecture 11
Real-Time Operating Systems

Feng-Li Lian
NTU-EE

Feb10 – Jun10

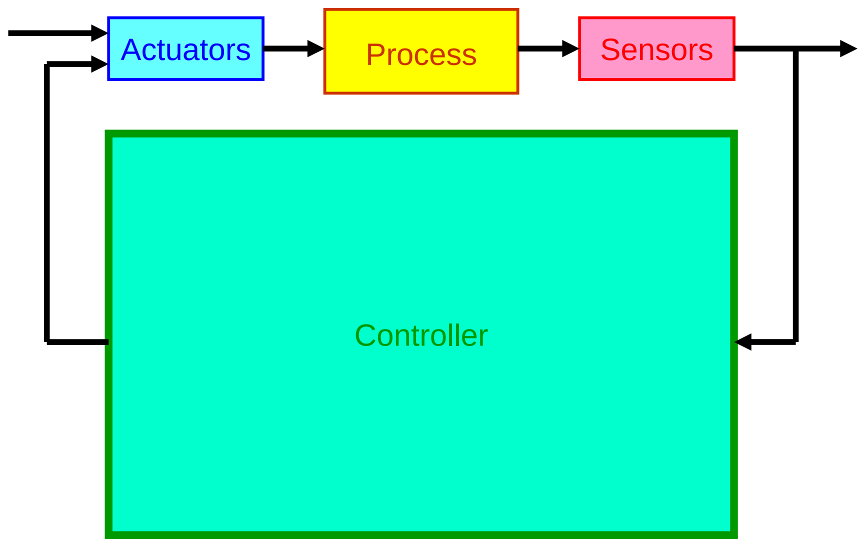
- Real-Time Control Systems
  - Controlled by one Computer Processor
    - Centralized control systems
    - Real-time operating systems
  - Controlled by one Communication Medium
    - Distributed control systems
    - Real-time communications



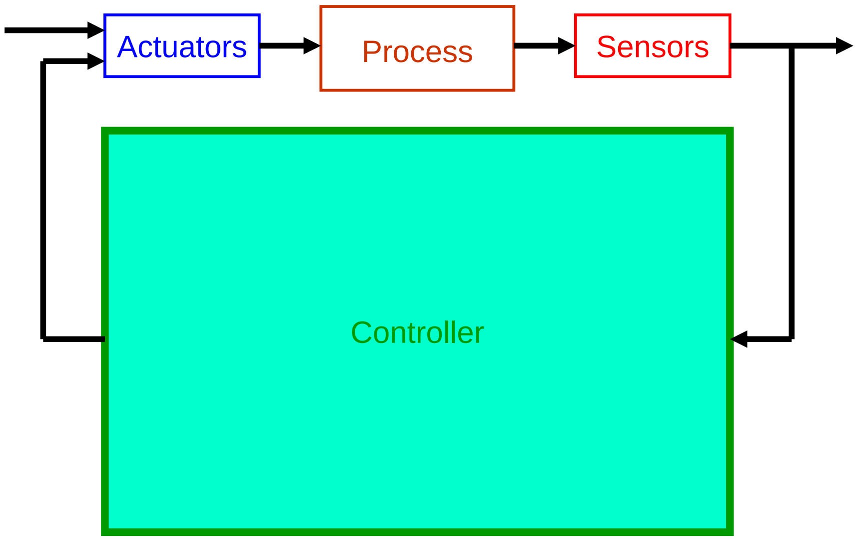
Centralized Control System

Distributed Control System

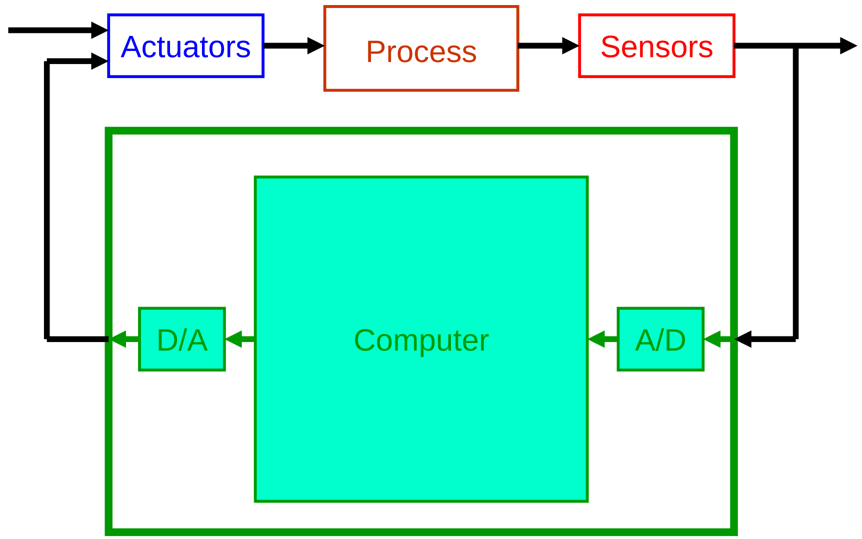
- Structure of Real-Time Control System
- Operating Systems
- Computer Control Systems
- Real-Time Computing
- Real-Time Operating Systems
- Task/Message/Packet Classification
- Hardware Requirements for Real-Time Applications



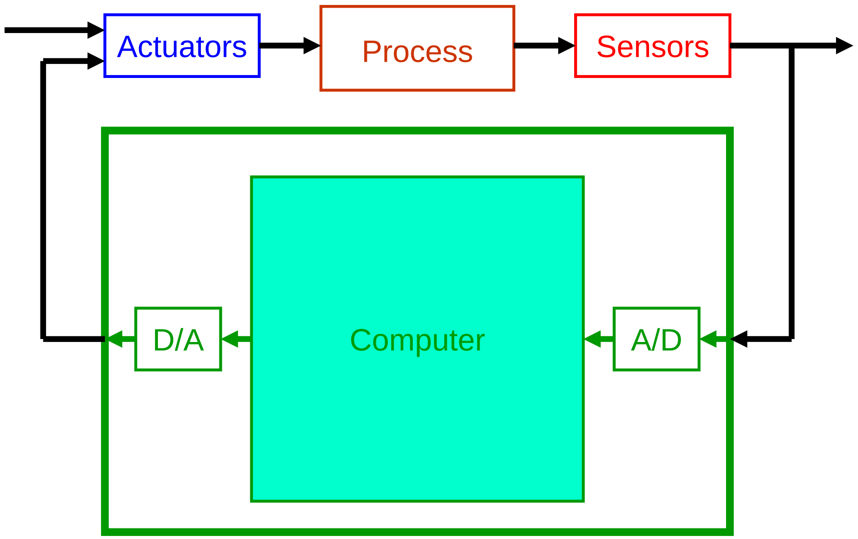
Structure of a Real-Time Control System



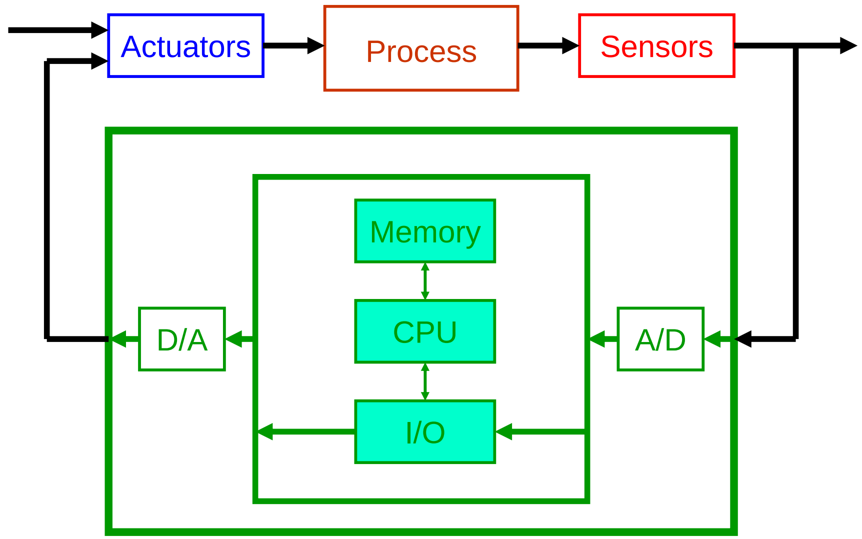
Structure of a Real-Time Control System

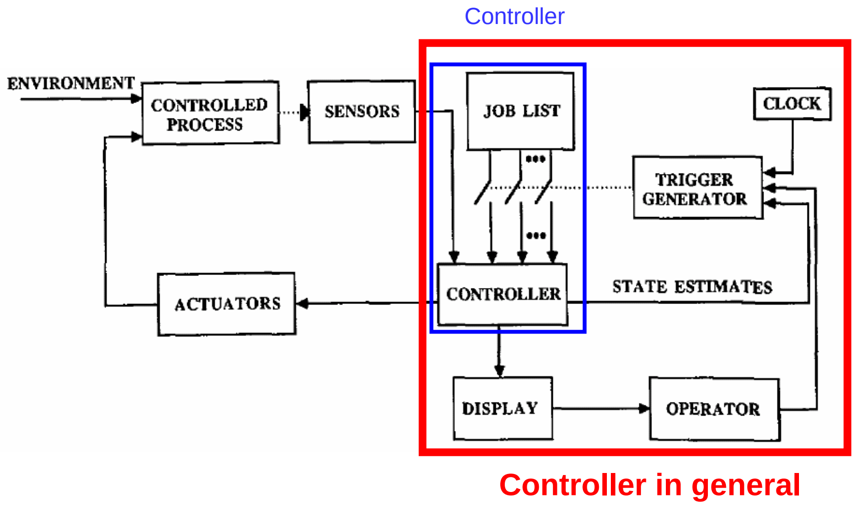


Structure of a Real-Time Control System



Structure of a Real-Time Control System





- Structure of Real-Time Control System
- Operating Systems
- Computer Control Systems
- Real-Time Computing
- Real-Time Operating Systems
- Task/Message/Packet Classification
- Hardware Requirements for Real-Time Applications

- Operating Systems (OS)
  - An operating system for a given computer converts the hardware of the system into a virtual machine with characteristics defined by the operating system
  - Operating systems are developed to support both real-time systems and multi-access on-line systems
    - General purpose OS
      - > A monolithic monitor
    - Minimal OS
      - > With a minimal kernel or nucleus
      - > For small, embedded applications

- General Purpose Operating System

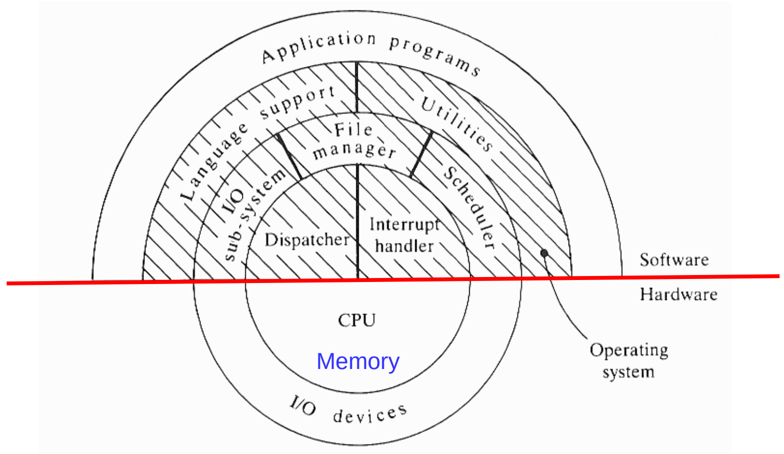


Figure 6.1 General purpose operating system.

▪ Minimal Operating System

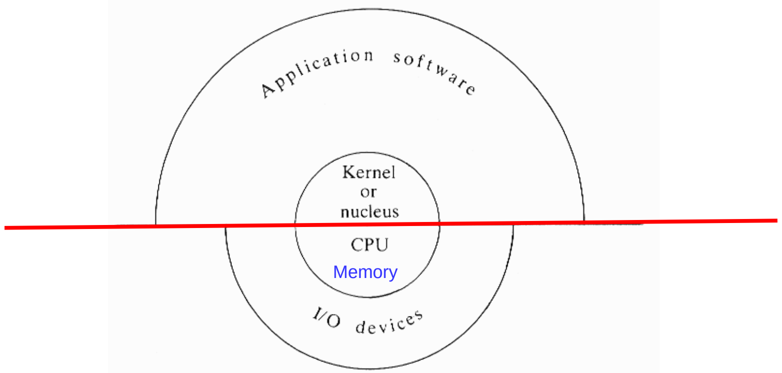


Figure 6.2 Minimal operating system.

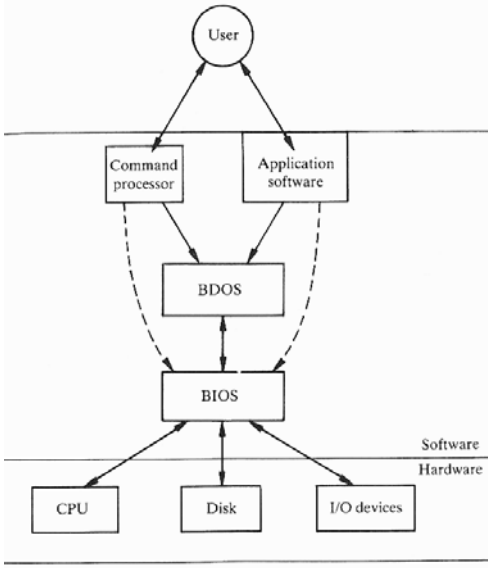


Figure 6.3 General structure of a simple operating system.

- BDOS:
  - Basic Disk OS
  - Handle input-output & file operations on disks
- BIOS:
  - Basic Input Output System
  - Various device drivers manipulate physical devices
- Access to OS is by means of subroutine calls
- Information is passed in CPU registers of machine
- Isolation between functions & high-level languages
- Connection between language and OS is made by compiler

▪ Multi-User or Multi-Programming Operating Systems

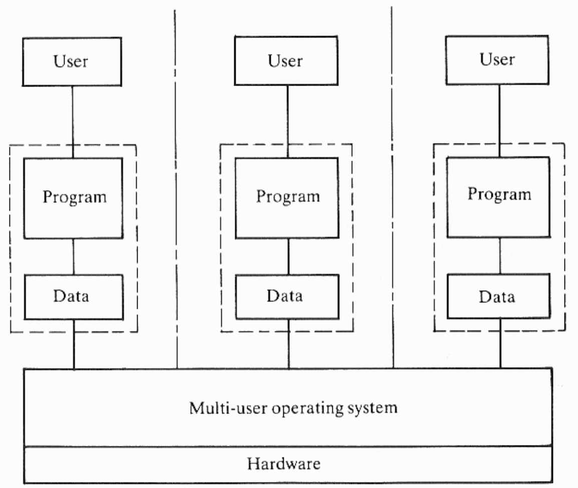


Figure 6.4 Multi-user operating system.

- User programs are run in own protected environment with no interference with others

▪ Multi-Tasking Operating Systems

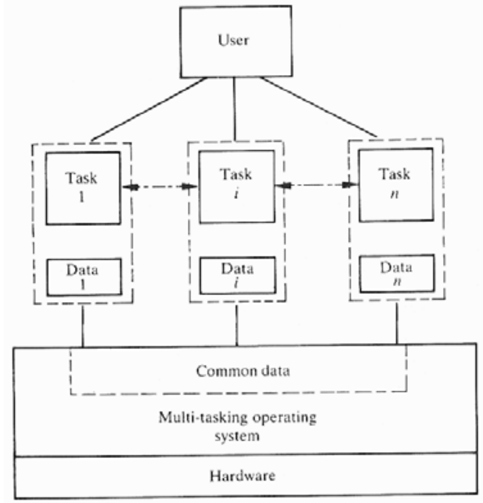


Figure 6.5 Multi-tasking operating system.

- A single user & various tasks cooperate
- Task communication
- Data sharing

▪ Real-Time Multi-Tasking Operating Systems

- Support resource sharing & timing requirements
- Functions:
  - Task management:
    - Allocation of memory & processor time (scheduling) to tasks
  - Memory management:
    - Control of memory allocation
  - Resource control:
    - Control of all shared resources other than memory & CPU time
  - Inter-task communication & synchronization:
    - Provide safe communication between tasks
    - Enable tasks to synchronize their tasks
- Standard Features:
  - For disk files, basic input/output device drivers, utility programs

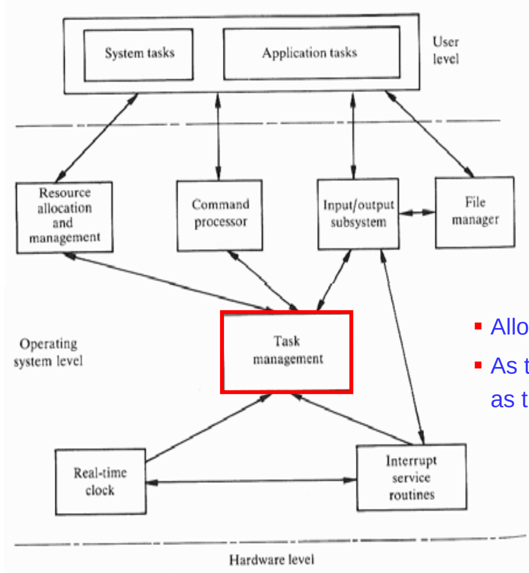


Figure 6.6 Typical structure of a real-time operating system.

- Allocating of the use of the CPU
- As the monitor or as the executive (control program)

- How to create a task
- What scheduling strategy the RTOS supports

- Structure of Real-Time Control System
- Operating Systems
- Computer Control Systems
- Real-Time Computing
- Real-Time Operating Systems
- Task/Message/Packet Classification
- Hardware Requirements for Real-Time Applications

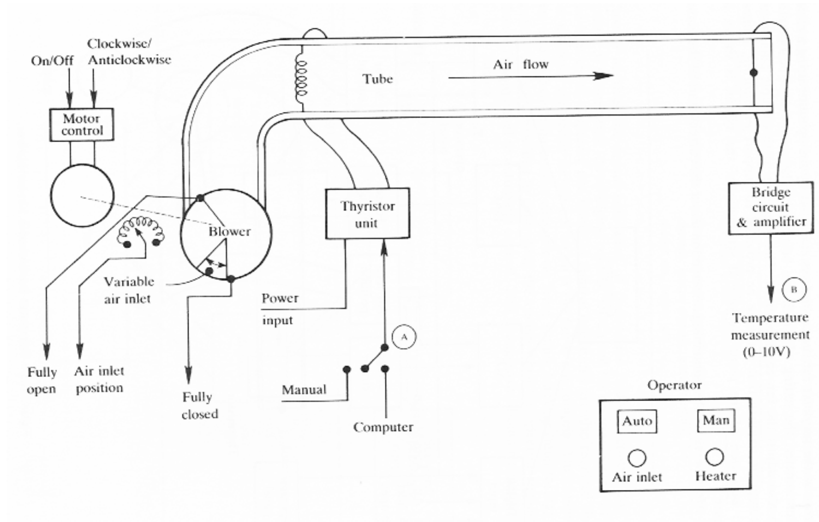


Figure 1.2 A simple plant – a hot-air blower.

Computer Control Systems

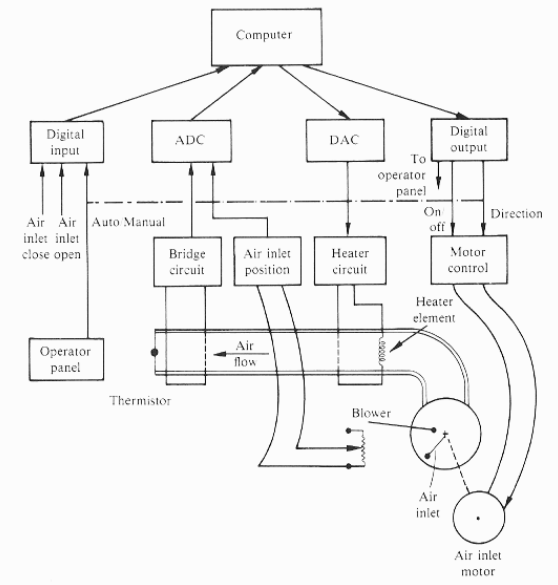


Figure 1.3 Computer control of a hot-air blower.

- Monitoring
  - Analog signals
  - Digital (logic) signals
- Control
  - Temperature control
  - Position control
  - Sequence control
- Actuation
  - Drive heat control
  - Move fan cover

Computer Control Systems: Generalized Computer System

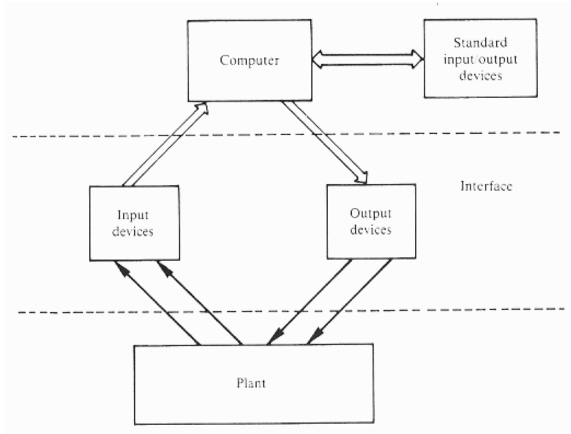


Figure 1.4 Generalised computer system.

Computer Control Systems: Hardware & Software Interface

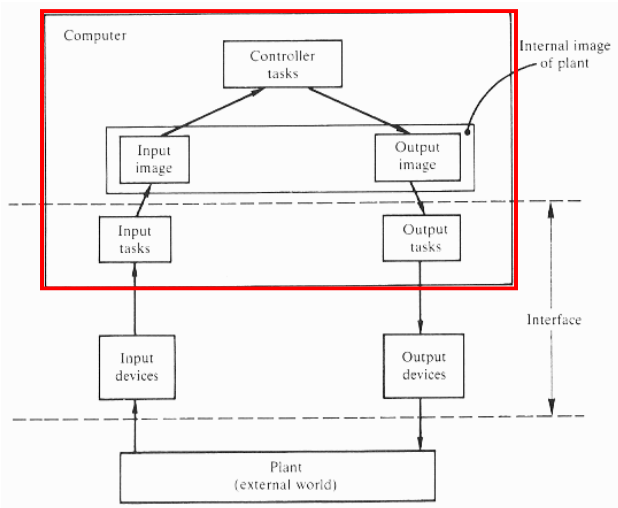


Figure 1.5 Generalised computer control system showing hardware and software interface.

- Sampling
- Digitalization

Computer Control Systems: With Communication Tasks

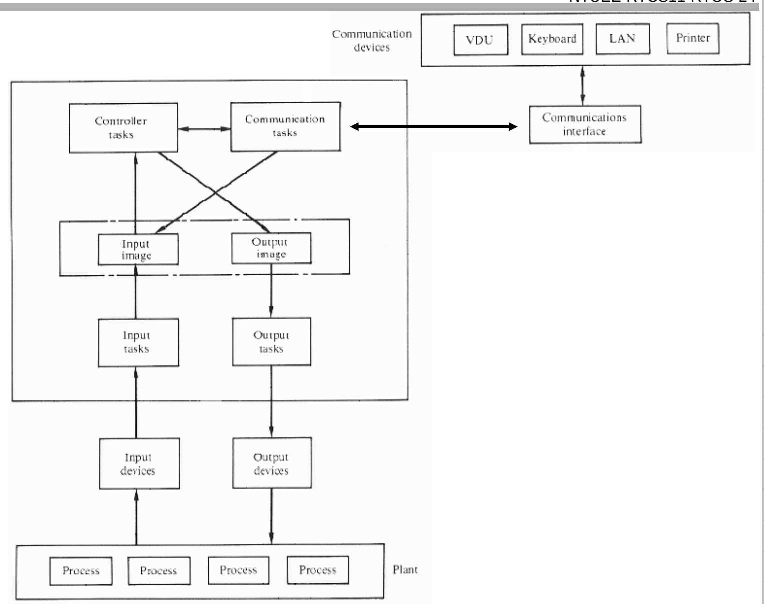


Figure 1.8 Computer control system showing communication tasks.

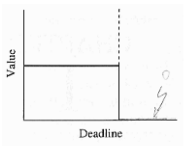
- Structure of Real-Time Control System
- Operating Systems
- Computer Control Systems
- Real-Time Computing
- Real-Time Operating Systems
- Task/Message/Packet Classification
- Hardware Requirements for Real-Time Applications

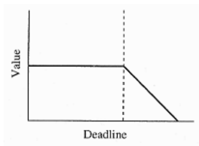
- Three major components:
  - Time:
    - Tasks must be assigned, scheduled, & completed before the deadline
    - Messages are required to be sent & received in a timely manner
    - The correctness of a computation depends not only on the logical correctness but also on the time at which the results are produced
  - Reliability:
    - Failure of a real-time system could cause an economical disaster or loss of human lives
  - Environment:
    - Where a computer operates

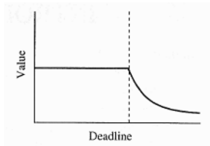
- Key features:
  - A real-time application:
    - Is usually comprised of a set of cooperating tasks
  - The tasks:
    - Invoked/activated at regular intervals
    - Have deadlines
  - In each invocation, a task
    - Senses the state of the system,
    - Performs certain computation
    - Sends command to change and/or display the state of system
  - e.g., an automobile application

- For example:
  - ABS: an automobile application:
    - A task may sense the pressure from the brake pedal and the speed of the individual wheels,
    - Perform computation to determine if a wheel is locked, and
    - Activate antilock braking actions by changing the position of the valves
  - Engine control: An aircraft-control application:
    - A task may monitor the current position of the throttle,
    - Perform computation based on the sensed position, and
    - Change thrust of engine by altering the fuel injected to it

- Hard Real-Time Systems:
  - Consequences of not meeting deadline is catastrophic
  - e.g., aircraft, nuclear reactors, chemical power plants, etc.
- Firm Real-Time Systems:
  - Result cease to useful as soon as deadline expires, but consequences of not meeting deadline not sever
  - e.g., transactions in a database system
- Soft Real-Time Systems:
  - Utility of results decreases over time after deadline expires
  - e.g., multimedia, temperature control, etc.







- Where do the deadlines come from?
- How does one know whether a deadline is hard, firm or soft?
- The deadlines come from applications
  - Automobile
  - Air-defense system
  - ATM, multimedia, temperature control, etc.

- Predictability:
  - 100% guarantee
    - Need to know exact characteristics of all task a priori
    - Periodic tasks with hard deadlines
  - Probabilistic guarantee
    - A certain fraction of tasks guaranteed to meet constraints
    - A given task has a certain probability of meeting constraints
  - Run-time deterministic guarantee
    - When a task is activated, the system determines whether or not the task's constraints can be satisfied without jeopardizing the guarantees provided to other tasks
    - If YES, provide 100% guarantee
    - If NO, reject the task
    - Dynamical arriving aperiodic tasks or dynamic load sharing

- Structure of Real-Time Control System
- Operating Systems
- Computer Control Systems
- Real-Time Computing
- Real-Time Operating Systems
- Task/Message/Packet Classification
- Hardware Requirements for Real-Time Applications

Real-Time Operating Systems

- 4 main functions:
  - Process management and synchronization
  - Memory management
  - Inter-process communication (IPC)
  - Input and Output (I/O)
- Must stress predictability and support real-time constraints
- 3 general categories of RTOS:
  - Small proprietary (homegrown & commercial) kernels
  - RT extensions to UNIX and others
  - Research kernels

Proprietary Kernels:

- Small & fast commercial RTOSs:
  - Such as QNX, PDOS, pSOS, VxWorks, Nulceus, ERCOS, EMERALDS, Windows CE
  - Fast context switch & fast interrupt response
  - Small size
  - No virtual memory & can lock code & data in memory
  - Multitasking & IPC via standard, well-known primitives such as mailboxes, events, signals, & semaphores

Proprietary Kernels:

- How to support real-time constraints
  - Bounded primitive execution time
  - Real-time clock
  - Priority scheduling
  - Special alarms and timeouts
  - Support RT queuing disciplines
  - Provide primitives to delay processing or suspend/resume execution

RT Extensions:

- RT-UNIX, RT-LINUX, RT-MACH, RT-POSIX
  - Slower, less predictable, but more functions & better development environments
  - Based on a set of familiar interfaces (standards)
  - RT-POSIX (Portable Operating System Interface):
    - 11 RT-related functions:
      - Timers, priority scheduling, RT files, semaphore, IPC, asynchronous event notification, process memory locking, threads, asynchronous and sync I/O

RT Extensions:

- RT-UNIX, RT-LINUX, RT-MACH, RT-POSIX
  - Inappropriate assumptions:
    - Optimize for the average case (not worst case)
    - Assign resources on demand
    - Ignore most of information about application
    - Schedule CPU and allocate resource independently that might cause unbounded blocking

Research Operating Systems:

- Support RT scheduling algorithms and timing analysis
- Develop RT sync primitives, e.g., priority ceiling
- Emphasize predictability over average performance
- Support for fault-tolerance and I/O

- Example:
  - Spring, MARS, HARTOS, MARUTI, ARTS, CHAOS, DARK

RT Languages and Some Experimental Ones:

- Ada, Modula-2,

- Flex: (Univ. of Illinois)
- Euclid: (Univ. of Toronto)

- Structure of Real-Time Control System
- Operating Systems
- Computer Control Systems
- Real-Time Computing
- Real-Time Operating Systems
- Task/Message/Packet Classification
- Hardware Requirements for Real-Time Applications

- Need:
  - Specification languages and
  - Performance measures:
    - Capable of expressing timing requirements
  - Means:
    - To predict the execution times of programs
    - To model the reliability of software and of hardware
    - To assign tasks to processors, and
    - To schedule them so that deadlines are met
  - Mechanisms:
    - System can quickly recover from the failure of an individual component

Krishna & Shin 97

- Critical or non-critical:
  - Depending on its function and system state
- Based on invocation behavior:
  - periodic
  - aperiodic
  - sporadic
- How do we derive message/packet deadlines?

Krishna & Shin 97

- Clock-based tasks:
  - Cyclic, periodic
  - Process time constant (characteristics)
  - Sampling time (rate)
- Event-based tasks:
  - Aperiodic, sporadic
  - In response to some event
- Interactive systems:
  - ATM, hotel reservation, car rental
  - In response to some state

Bennett 94

- Structure of Real-Time Control System
- Operating Systems
- Computer Control Systems
- Real-Time Computing
- Real-Time Operating Systems
- Task/Message/Packet Classification
- Hardware Requirements for Real-Time Applications

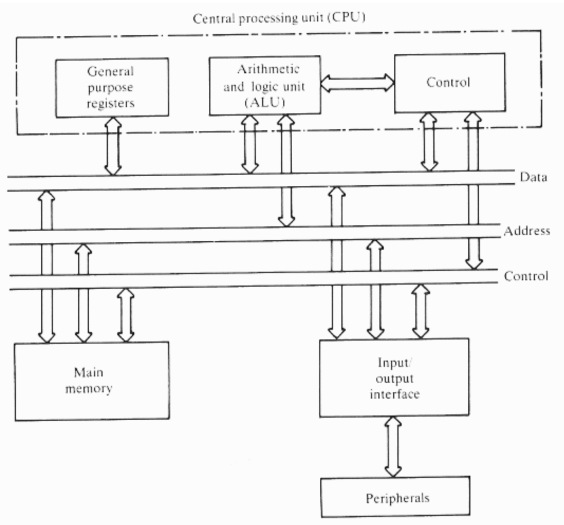
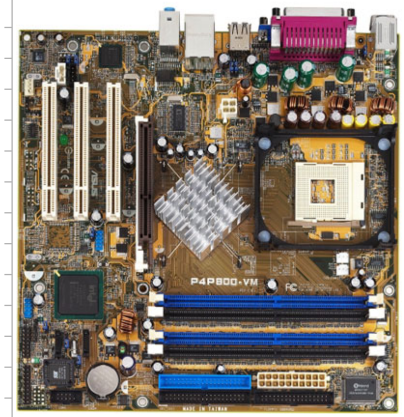
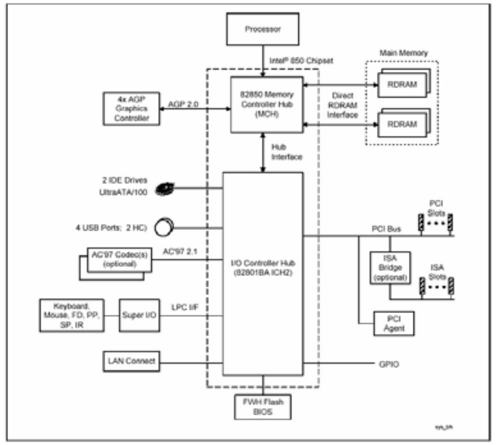


Figure 3.1 Schematic diagram of a general purpose digital computer.





ASUS P4P800-VM,
Intel® Pentium® 4 Processor in the 478-pin Package/Intel® 850 Chipset Family Platform, 2003

- Computers:
  - Microprocessors:
    - Intel XX86, Motorola 680XX, National 32XXX, Zilog Z80 & Z8000, etc.
  - Microcontrollers:
    - Motorola MPC 555/556, etc.
  - Specialized digital signal processors
    - Fast DSP, parallel computers, etc.

- Key Components in a Computer:
  - Central Processing Unit (CPU)
  - Storage/Memory
  - Input/Output Device

Central Processing Unit (CPU):

- Arithmetic and logic unit (ALU)
  - Arithmetic and logic operations:
    - Add, subtract and compare numbers
  - Multiplication and division is provided by other hardware units
  - Floating point arithmetic unit
- General purpose registers
  - Store data temporarily
- Control unit
  - Supervise operations within CPU
  - Fetch program instructions from main memory
  - Decode instructions
  - Set up data paths and timing cycles for execution of instructions

Central Processing Unit (CPU):

- Features:
  - Wordlength
    - For precision in calculation and direct access to main storage within one instruction word
  - Instruction set
    - Features to reduce the number of instructions required to perform "housekeeping" operations, reduce storage requirements, and improve operation speed
  - Addressing methods
  - Number of registers
  - Information transfer rates
    - Within CPU and between backing store & CPU, with I/O devices
  - Interrupt structure

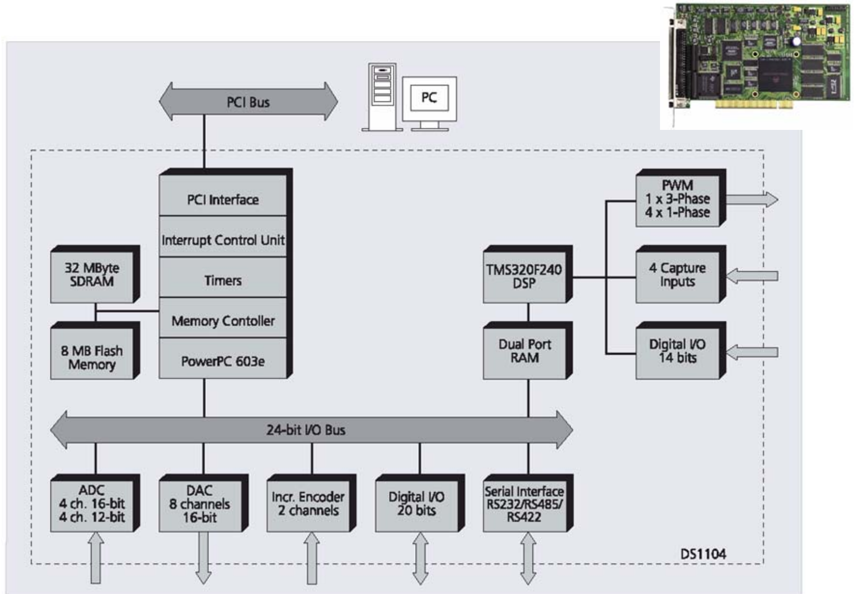
Storage/Memory:

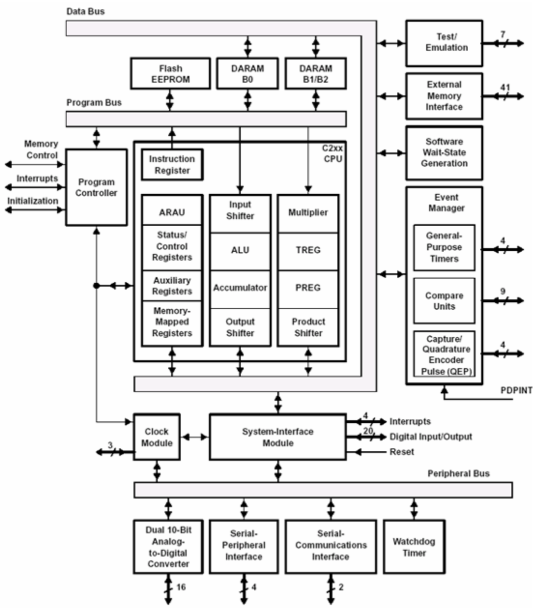
- Fast access storage:
  - RAM (random access memory – read/write)
  - ROM (read-only memory)
    - Prevent loss due to power failure or malfunctioning
  - PROM (programmable ROM) by ROM burners
    - Factory-programmed ROM (mask-programmed ROM)
    - Field-programmable ROM
  - EPROM (electronically or erasable programmable ROM)
    - UV-EPROM: Erased by ultraviolet light
    - Flash PROM: Erased by standard system voltage
- Auxiliary storage:
  - Disk
  - Magnetic tape

Input & Output (I/O):

- Sections:
  - Process I/O
  - Operation I/O
  - Computer I/O
- Features:
  - Parallel or serial data transfers
  - Analog-to-digital or digital-to-analog conversion (ADC/DAC)
  - Conversion to pulse rates

- Single-Chip Microprocessors & Microcontrollers:
  - Microprocessors:
    - + EPROM + RAM + Oscillator
    - + Hardware Timers + Interrupt Controller
    - + Serial Communication Controller + I/O Ports
    - (+ external memory chip)
  - Microcontrollers:
    - Microprocessor
    - + multiplexed ADC + process output (e.g., PWM)
    - + real-time clock generator + watch-dog timer





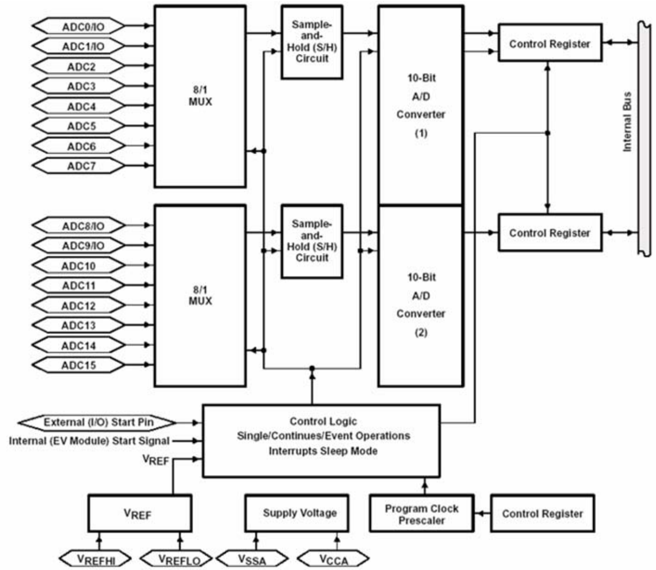


Figure 12. Analog-to-Digital Converter Module

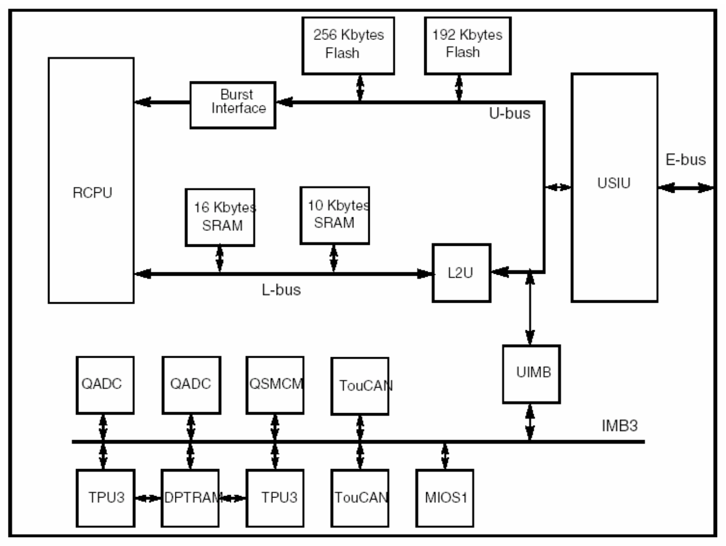


Figure 1-1 MPC555 / MPC556 Block Diagram

- Specialized Processors:
  - Safety-critical applications
    - To simplify the instruction set
      - RISC (reduced instruction set computer)
  - Increased computation speed
    - Parallel computer architecture
    - Digital signal processors
  - RISC:
    - For formal verification of processing logic
    - Easier to write assemblers & compilers
    - VIPER:
      - Formal math description of processor logic
      - Integer arithmetic (32 bit) and no floating point operations
      - No interrupts – using polling
      - No dynamic memory allocation

- Process-Related Interfaces:
  - Digital quantities
    - Binary, generalized digital quantity (binary coded decimal)
  - Analog quantities
    - e.g., Thermocouple, strain gauge, voltage, current
  - Pulses & pulse rates
    - A series of pulse of fixed duration
    - A single pulse of variable length
    - Pulse width modulation (PWM)
  - Telemetry
    - Remote stations

- Data Transfer Techniques:
  - Polling
    - Busy wait
    - Periodic check
  - Interrupts
    - Saving & restoring registers
    - Interrupt input mechanisms
    - Interrupt response mechanisms
    - Multi-level interrupts
  - Direct Memory Access
    - Burst mode: Full control over the data highways
    - Distributed mode: Occasional control
    - Cycle stealing : When CPU is not using the data bus

▪ Communications:

- Asynchronous & synchronous transmission techniques
- Local- & wide-area networks