

CS4220 Embedded Systems CS6235 Real-Time Systems

1B: RTE Concepts and Examples

Instructor: Calton Pu

calton.pu@cc

TA: TBA (???)@cc)



Example: Car

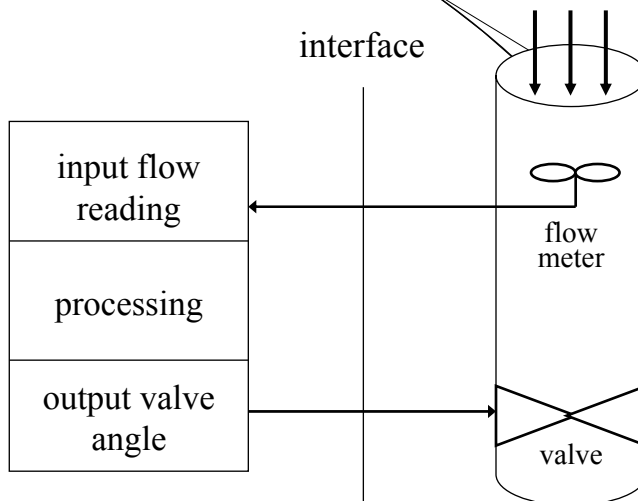
- Operating environment: Road conditions and other cars.
- Controlling System
 - Human driver: Sensors - Eyes and Ears of the driver.
 - Computer: Sensors - Cameras, Infrared receiver, and Laser telemeter.
- Controls: Accelerator, Steering wheel, Break-pedal.
- Actuators: Wheels, Engines, and Brakes.

Example: cruise control

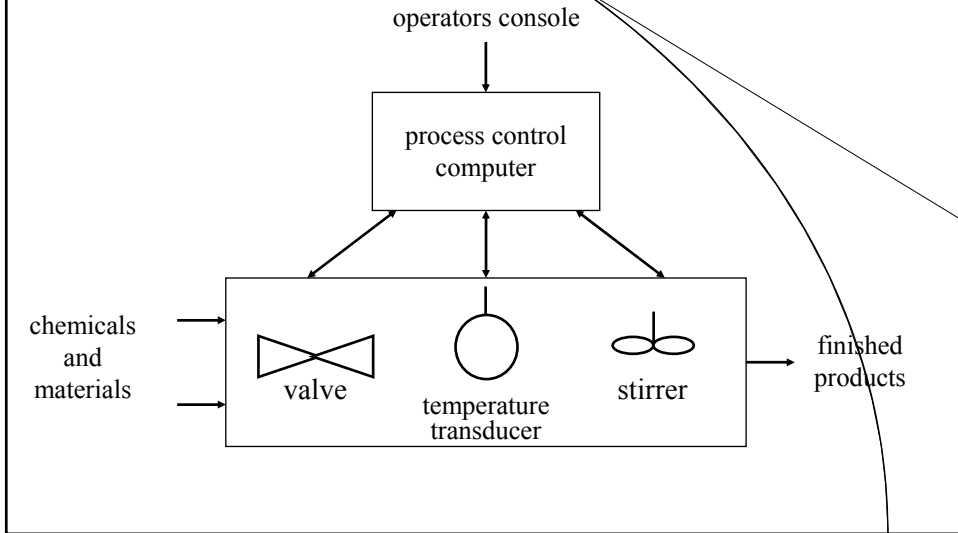
- Regulates speed of car by adjusting the throttle: driver sets a speed and car maintains it.
- Measures speed through device connected to drive shaft.
- Hard real-time: drive shaft revolution events.
- Soft real-time: driver inputs, throttle adjustments.



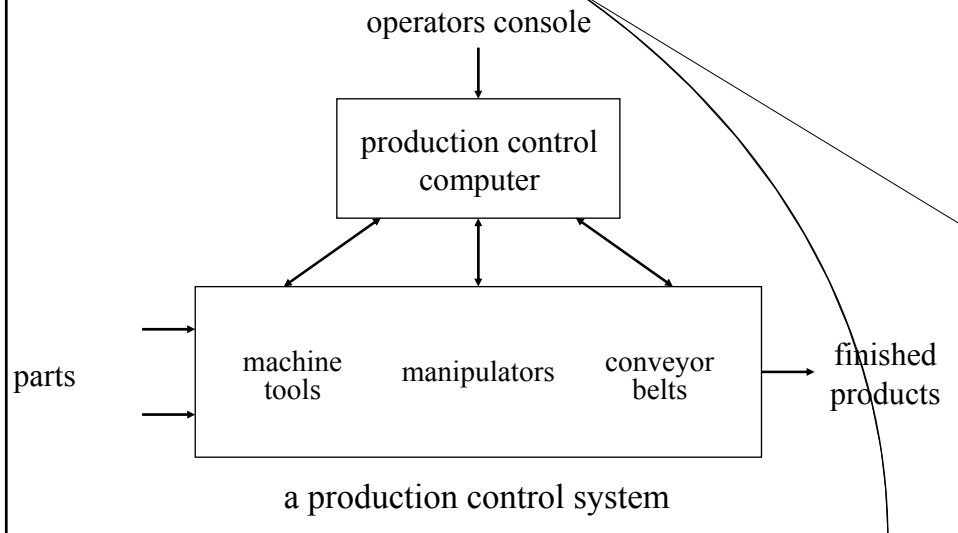
Simple Valve Control



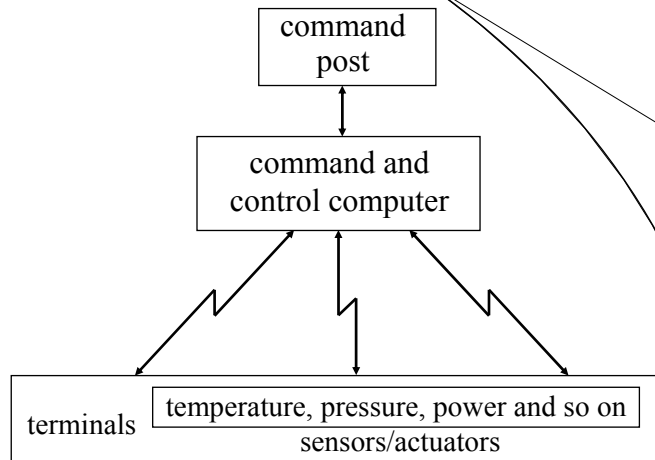
Process Control



Manufacturing

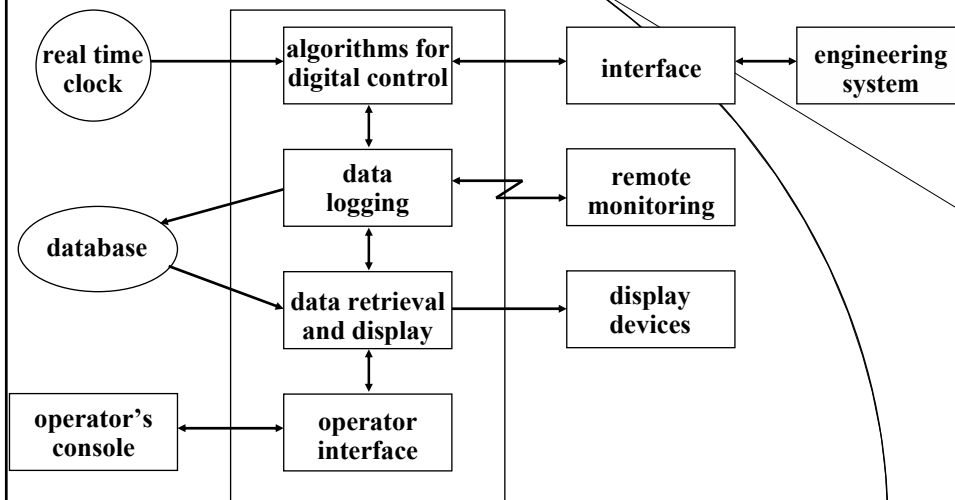


Command, Control, Communications

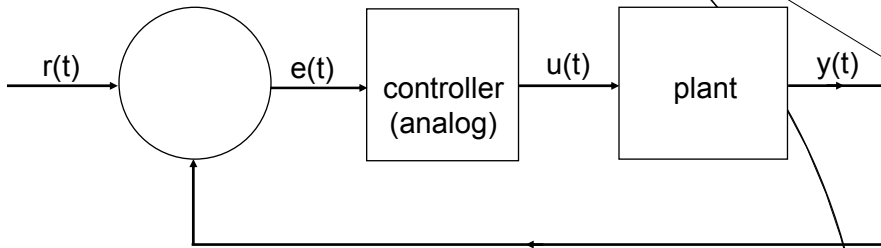


a command and control system

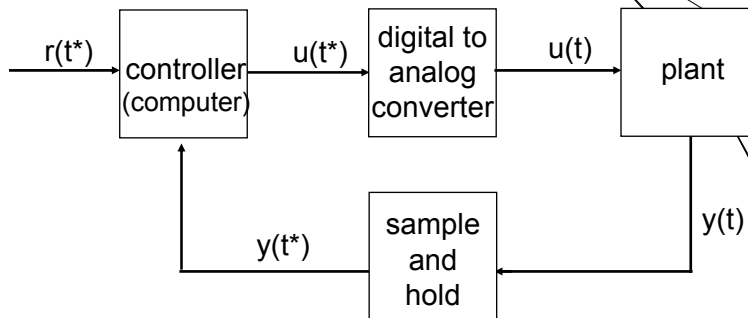
Industrial Embedded System



Feedback Control System



Digital Feedback Control



More examples

- cars: engine control, ABS, drive-by-wire
- planes: stability, jet engine, fly-by-wire
- computers: peripherals, applications
- military: weapons, satellites
- domestic: microwave, thermostat, dishwasher
- medical: pacemaker, medical monitoring
- protection: intruder alarm, smoke/gas detection

RT Systems Terminology

- System: black box with n inputs and m outputs
- Response time: time between presentation of a set of inputs and the appearance of the corresponding outputs
- Utilization: measure of 'useful' work a system performs
- Events: Change of state causing a change of flow-of-control of a computer program

Classification of RT Systems

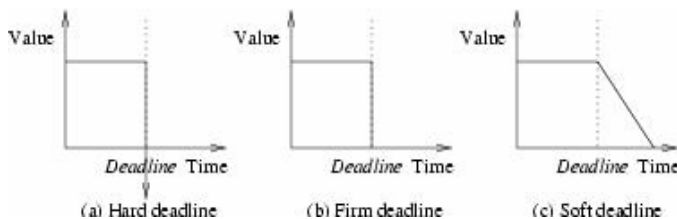
- synchronous: events occur at predictable times in the flow-of-control.
- asynchronous: interrupts.
- state-based vs. event-based:
 - plane wing is at an angle of 32° (state)
 - plane wing moved up 4° (event)
- deterministic system: for each possible state and each set of inputs, a unique set of outputs and next state of the system can be determined.

More RT Terminology

- RTS: Correctness depends on results PLUS the time of delivery! Failure can have severe consequences.
- What are real-time systems? Planes, cars, washer, video player, thermostat, video games, weapons,...
- Related: QoS management, resource management, adaptive systems, embedded systems, pervasive and ubiquitous computing, ...

RT Systems Classification (2)

- **HARD:** miss a deadline and you're in trouble! (planes, trains, factory control, nuclear facilities, ...)
- **SOFT:** try to meet deadlines, but if not, system still works, although with degraded performance (multimedia, thermostat, ...)
- **FIRM:** late results are worthless, but you are not in trouble



Characteristics of RT Systems

- size: small assembler code or large C++, Ada, ... code (example: 20 million lines of Ada for Intl. Space Station).
- concurrent control of separate components (model this parallelism with parallelism in your program).
- use of special purpose hardware and tools to program devices for this hardware in a reliable manner.

Common Misconceptions

- “real fast” is real-time: a computer system may satisfy an application’s requirement, but no predictability (no real-time resource management).
- hardware over-capacity is enough: again, without real-time resource management no appropriate balance of resource distribution.

Static Predictability

- RT system: satisfying the time constraints
 - Certain assumptions about workload and sufficient resource availability
 - Certify at “design time” that all the timing constraints of the application will be met
- For static systems, 100% guarantees can be given at design time
 - Immutable workload and system resources
 - System must be re-certified if anything changes

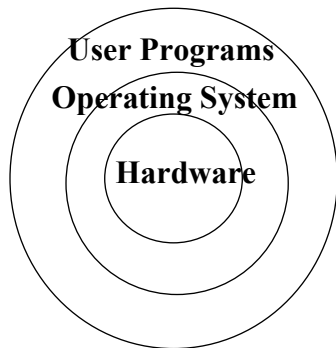
Dynamic Predictability

- Dynamic systems: not statically defined
 - Changeable system configuration
 - Changeable workload
- Dynamic predictability
 - Under appropriate assumptions (sufficient resources)
 - Tasks will satisfy time constraints

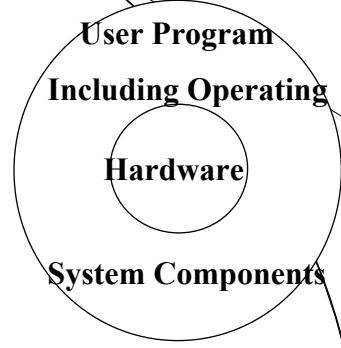
Reliability

- Reliability
 - Randell et al (1978)
“a measure of the success with which the system conforms to some authoritative specification of its behavior“
- Safety and reliability often interchangeable
 - Usually expressed in probabilities
- Other frequently used term: dependability.

Role of Operating Systems



Typical OS Configuration



Typical Embedded Configuration

Real-Time OSs

- Real-Time OS: VxWorks, QNX, LynxOS, eCos, DeltaOS, PSX, embOS, ...
- GPOS: no support for real-time applications, focus on 'fairness'.
- BUT, people love GPOSs, e.g., Linux:
 - RTLinux (FSMLabs)
 - KURT (Kansas U.)
 - Linux/RT (TimeSys)

RT OSs

- Determinism / Predictability
 - Ability to meet deadlines
 - Traditional operating systems non-deterministic
- Standards: Real-Time POSIX 1003.1
 - Pre-emptive fixed-priority scheduling
 - Synchronization methods
 - Task scheduling options

Lynx OS

- Lynx OS
 - Microkernel Architecture
 - Provides scheduling, interrupt, and synchronization support
 - Real-Time POSIX support
 - Easy transition from Linux

VxWorks

- Monolithic Kernel
 - Reduced run-time overhead, but increased kernel size compared to Microkernel designs
- Supports Real-Time POSIX standards
- Common in industry
 - Mars missions
 - Honda ASIMO robot
 - Switches
 - MRI scanners
 - Car engine control systems

RT Linux

- “Workaround” on top of a generic O/S
 - Generic O/S – optimizes average case scenario
 - RTOS – need to consider WORST CASE scenarios to ensure deadlines are met
- Dual-kernel approach
 - Makes Linux a low-priority pre-emptable thread running on a separate RTLinux kernel
 - Tradeoff between determinism of pure real-time O/S and flexibility of conventional O/S
- Periodic tasks only

RT Concepts

- Concurrency
- Scheduling: priorities, time driven, event driven, task scheduling (RMS).
- Processes, threads.
- Synchronization: test-and-set instructions, semaphores, deadlocks (circular waits), ...

RT Scheduling

- static: all scheduling decisions are determined before execution.
- dynamic: run-time decisions are used.
- periodic: processes that repeatedly execute
- aperiodic: processes that are triggered by asynchronous events from the physical world.
- sporadic: aperiodic processes w/ known minimum inter-arrival jitter between any two aperiodic events.

Preemptive vs. Non-preemptive

- **Preemptive Scheduling**
 - Task execution is preempted and resumed later.
 - Preemption takes place to execute a higher priority task.
 - Offers higher schedulability.
 - Involves higher scheduling overhead due to context switching.
- **Non-preemptive Scheduling**
 - Once a task is started executing, it completes its execution.
 - Offers lower schedulability.
 - Has less scheduling overhead because of less context switching.

Rate Monotonic Priority Assignment (RMA)

- each process has a unique priority based on its period; the shorter the period, the higher the priority.
- RMA proven optimal in the sense that if any process set can be scheduled (using preemptive priority-based scheduling) with a fixed priority-based assignment scheme, then RMA can also schedule the process set.

RMA

- Each task has a period T and run-time C .
- System utilization $U = \sum(C_i/T_i)$. Measure for computational load on the CPU due to the task set.
- There exists a maximum value of U , below which a task set is schedulable and above which it is not schedulable.
- RMA: Liu and Layland (1973):
$$\sum(C_i/T_i) \leq n(2^{1/n} - 1)$$

Real-Time Languages

- Support for the management of time
 - Language constructs for expressing timing constraint, keeping track of resource utilization.
- Schedulability analysis
 - Aid compile-time schedulability check.
- Reusable real-time software modules
 - Object-oriented methodology.
- Support for distributed programming and fault-tolerance

Real-Time Databases

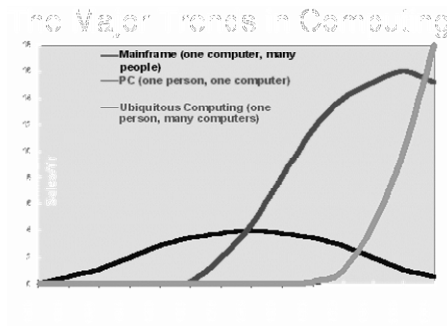
- Most conventional database systems are disk-based.
- They use transaction logging and two-phase locking protocols to ensure transaction atomicity and serializability.
- These characteristics preserve data integrity, but they also result in relatively slow and unpredictable response times.

Real-Time Databases (2)

- In a real-time database system, important issues include:
 - transaction scheduling to meet deadlines.
 - explicit semantics for specifying timing and other constraints.
 - checking the database system's ability of meeting transaction deadlines during application initialization.

New Environments

- **Ubiquitous Computing:** make computers invisible, so embedded, so fitting, so natural, that we use it without even thinking about it.



One Vision of Future

- **Autonomous Computing:**
 - self-configurable
 - self-adapting
 - optimizing
 - self-healing
- **Building real-time systems:**
 - toolkits, validation tools, program composition
 - Boeing 777: \$4Billion, >50% system integration & validation!

New Constraints

- Soft real-time applications:
 - mainstream applications
 - notion of QoS
- Multi-dimensional requirements:
 - real-time, power, size, cost, security, fault tolerance
 - conflicting resource requirements and system architecture
- Unpredictable environments:
 - Internet (servers), real-time databases, ...

RT and Embedded Systems

- An “engineering approach” to RTE
 - Model of RTE systems (e.g., tasks with time constraints)
 - Techniques that satisfy the constraints (e.g., scheduling algorithms such as RMA)
 - Implementation of these techniques
- Uncertainties of today’s environments
 - Ubiquitous/pervasive/environmental computing