Introduction to Real Time systems

OUTLINE

- Real time requirements
- Digital control: PID
- Selection of the sampling period
- Control law computation timing
- Control hierarchy
- Examples of real time applications
  - signal processing
  - real time databases

Ref: [Liu]

- Ch. 1 (pg. 1 – 25)

Examples of real time applications

Guided bombs

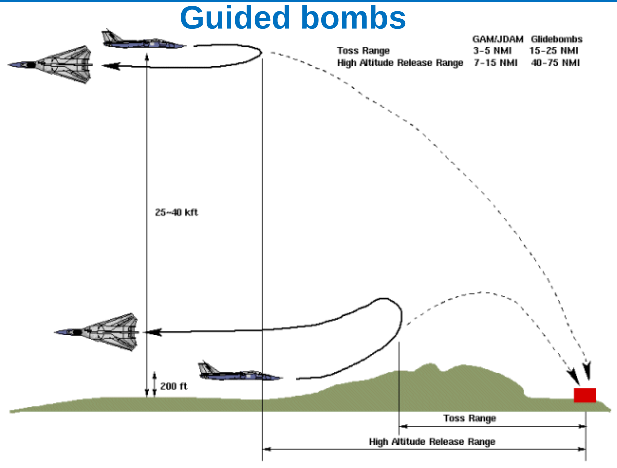
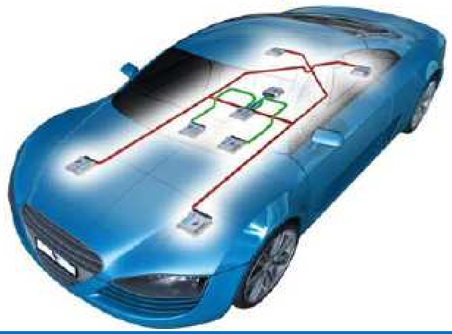

GAM/JDAM Glidebombs
Toss Range 3-5 NMI 15-25 NMI
High Altitude Release Range 7-15 NMI 40-75 NMI
25–40 kft
200 ft
Toss Range
High Altitude Release Range

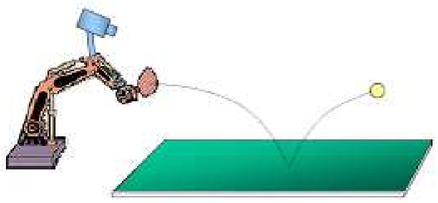
FIG.10 DELIVERY PROFILES FOR GPS/INERTIAL GUIDED WEAPONS

Flight control systems

Automotive systems

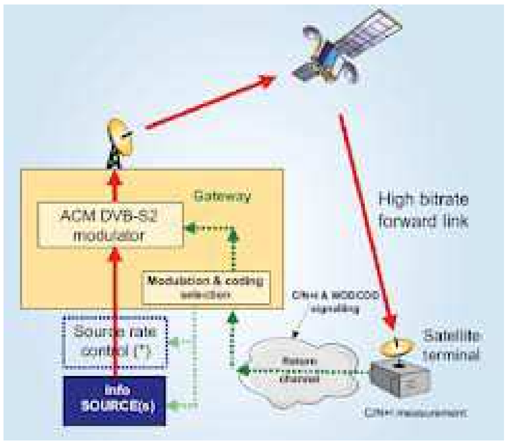


Ping-pong robot



Examples of real time applications

Digital video broadcasting



Curiosity Mars rover

Image processing: face detection

… and many more

Real-Time vs. Non-Real-Time Systems

Q: What distinguishes RT systems from non-RT systems?
A: Timing constraints!

Timing parameters:

- **Release Time (*r*):**	time when job becomes available for execution
- **Completion Time (*f*):**	time when job finishes execution
- **Response Time**:	(*f* - *r*)
- **Deadline (*d*):**	time when execution must be completed
- **Relative Deadline (*D*):**	maximum response time (*D* = *d* - *r*)

Timing constraints for Real-Time systems:

$\boldsymbol{f \le d}$ (completion time before deadline)

$\boldsymbol{f - r \le D}$ (response time shorter then relative deadline)

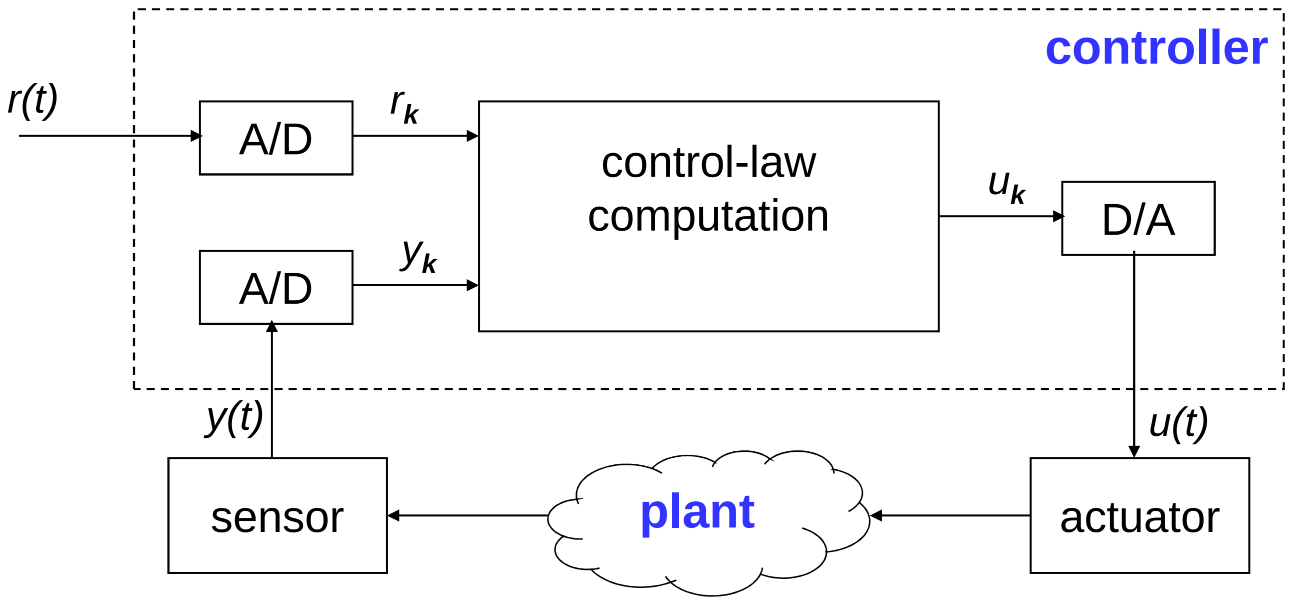
Typical Real-Time Applications

- Digital control: sampled data systems.
- High-level controls: planning and policy above low level control.
- Signal processing: digital filtering, radar signal processing.
- Real-time databases.
- Telecommunication systems.
- Multimedia applications: video compression & encryption.

Real-Time requirements

- Hard-real time systems: where there is a high penalty for missing a deadline
  - e.g., control systems for aircraft/space probes/nuclear reactors; refresh rates for video, or DRAM.
- Soft real-time systems: where there is a steadily increasing penalty if a deadline is missed.
  - e.g., laser printer: rated by pages-per-minute, but can take differing times to print a page (depending on the "complexity" of the page) without harming the machine or the customer;
  - however, the rate at which the image is delivered to the drum is a hard deadline, since the image will be lost otherwise.
- Many systems have hard-deadlines imposed by the operations of peripherals, e.g., DRAM (refresh rate), disk (spin rate), cameras (scan rate), displays (refresh rate).

Digital control: PID



$e(t)=r(t) - y(t)$

$u(t)=a\cdot e(t) + b\cdot e'(t) + c\cdot\int e(t)dt$

$e_k = r_k - y_k$

$u_k = f(e_k , e_{k-1} , e_{k-2} , e_{k-3} , \ldots)$

Digital control: PID (2)

$u(t)=a\cdot e(t) + b\cdot e'(t) + c\cdot\int e(t)dt$

$u_k = f(e_k , e_{k-1} , e_{k-2} , e_{k-3} , \ldots)$

$u(t)=a\cdot \mathbf{P}(t) + b\cdot \mathbf{D}(t) + c\cdot \mathbf{I}(t)$

$u_k = a\cdot \mathbf{P}_k + b\cdot \mathbf{D}_k + c\cdot \mathbf{I}_k$

at time t_k: $\mathbf{P}_K = e(t_k) = e_k$

$\mathbf{D}_k = e'(t) = de / dt \cong (e_k - e_{k-1}) / (t_k - t_{k-1}) = (e_k - e_{k-1}) / T_S$

(T_S is the sampling period)

$$\mathbf{I}_k = \int_0^{t_k} e(t)dt \cong \int_0^{t_{k-1}} e(t)dt + (e_k+e_{k-1})\cdot T_S/2 = \mathbf{I}_{k-1} + (e_k+e_{k-1})\cdot T_S/2$$

then: $u_k = a\cdot e_k + b\cdot(e_k - e_{k-1}) / T_S + c\cdot[\mathbf{I}_{k-1} + (e_k+e_{k-1})\cdot T_S/2]$

and also: $u_{k-1} = a\cdot e_{k-1} + b\cdot(e_{k-1} - e_{k-2}) / T_S + c\cdot \mathbf{I}_{k-1}$

$u_k - u_{k-1} = a\cdot(e_k - e_{k-1}) + b\cdot(e_k - 2e_{k-1} + e_{k-2}) / T_S + c\cdot(e_k + e_{k-1})\cdot T_S/2$

$u_k = u_{k-1} + \alpha\cdot e_k + \beta\cdot e_{k-1} + \gamma\cdot e_{k-2}$

Digital control: PID (3)

```
set timer to interrupt periodically with period $T_s$;
at each timer interrupt, do
    • perform analog-to-digital conversions to get $y_k$ and $r_k$;
    • compute $e_k$ and control output $u_k$ (using previously saved
      values of $u_{k-1}$, $e_{k-1}$, $e_{k-2}$):
            $e_k = r_k - y_k$
            $u_k = u_{k-1} + \alpha \cdot e_k + \beta \cdot e_{k-1} + \gamma \cdot e_{k-2}$;
    • save new values of $e_k$ and $u_k$:
            $e_{k-2} = e_{k-1}$
            $e_{k-1} = e_k$
            $u_{k-1} = u_k$;
    • output $u_K$ and perform digital-to-analog conversion;
end do;
```

Selection of sampling period

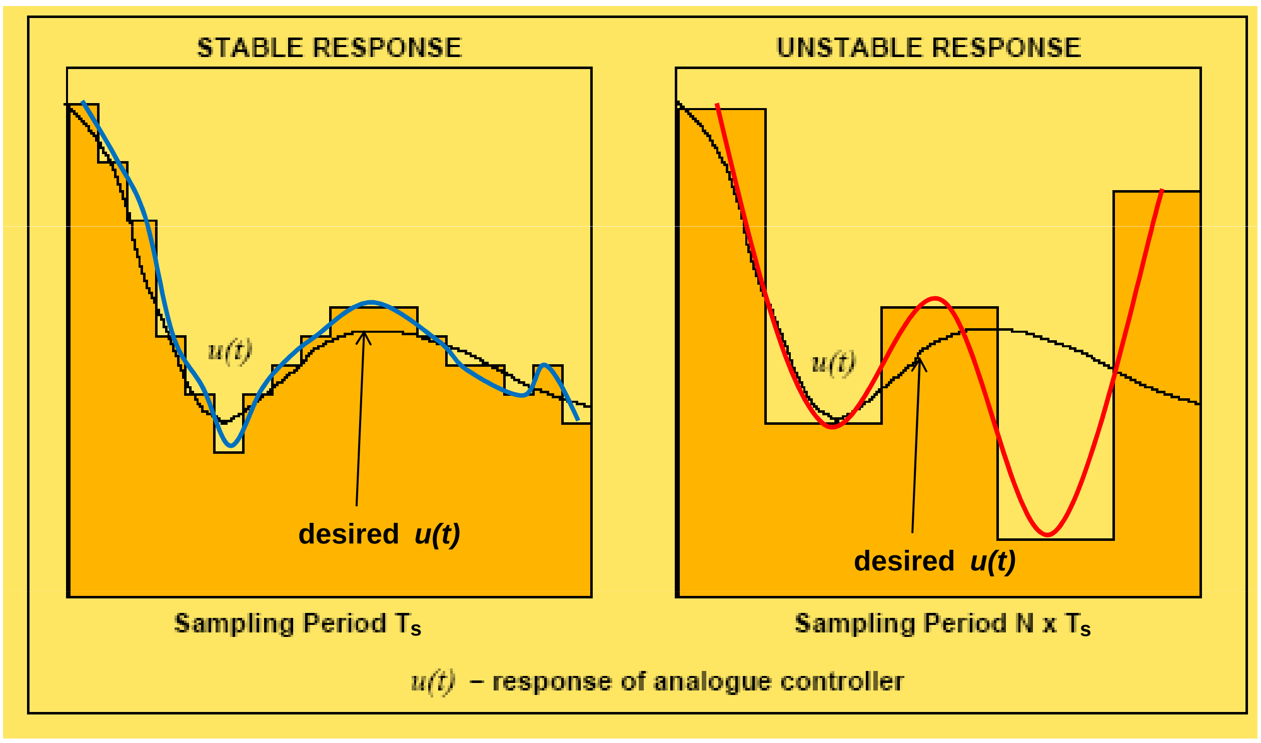
- The length T_s of time between any two consecutive instants at which the inputs are sampled is called the sampling period.
- T_s is a key design choice.
- The behavior of the digital controller critically depends on this parameter.
- Ideally we want the sampled data version to behave like the analogue controller version.
- This can be done by making T_s very small.
- However this increases the computation required.
- We need a good compromise.

Selection of sampling period (2)

We need to consider two factors:

1. The perceived responsiveness of the overall system
   - sampling introduces a delay in the system response.
   - a human operator may feel the system ‘sluggish’ if the delay in response to his input is greater than 100 ms
2. The dynamic behavior of the system

Selection of sampling period (3)



Selecting sampling period to ensure dynamic behavior of the system

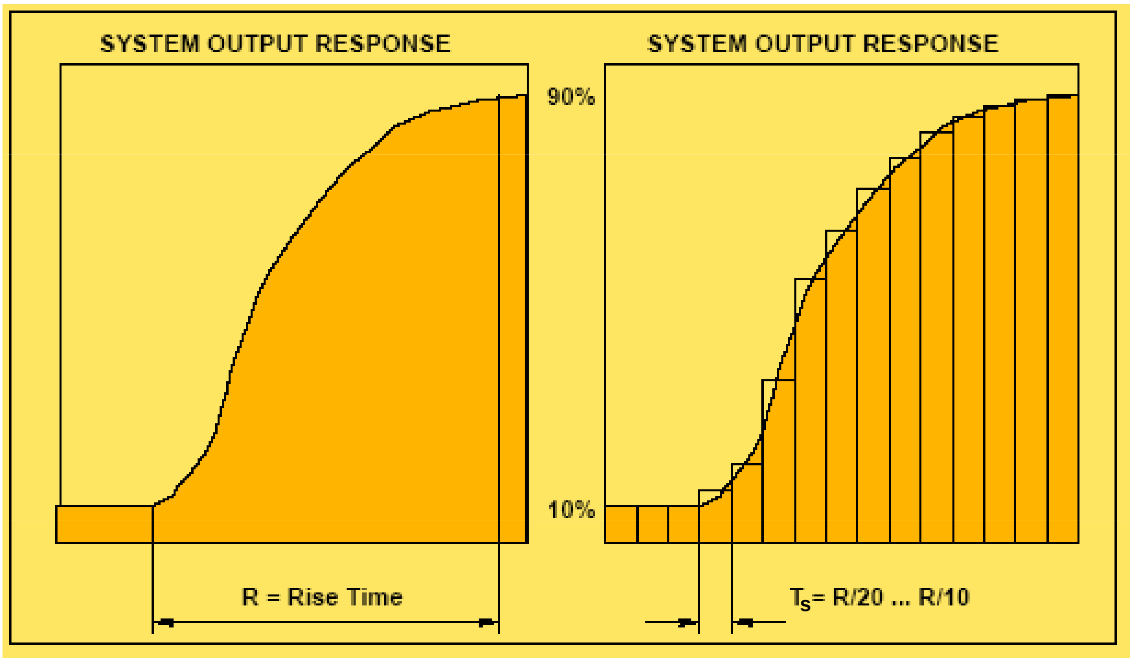
- In general, the faster a system can and must respond to changes, the shorter the sampling period should be.
- We can measure the responsiveness of the system by its rise time R.
- R is the time it takes to get close to its final state after the input changes (from 10% to 90% of the step).
- Typically you would like the ratio R/T_s of rise time to sampling period to be at least 10:

$$T_s \leq R/10$$

(i.e. at least 10 samples during rise time).

- A shorter sampling period is likely to reduce the oscillation in the system at the cost of more computation.

Sampling period T_s vs. Rise time



Sampling rate

- One can also refer the sampling rate ($1/T_s$) to the bandwidth B of the signal to be sampled.
- B is related to rise time R by the following formula: $B \cdot R \cong 0.5$ (it's $\cong 0.35$ for a simple one stage low pass *RC* network).
- To have $T_s \leq R/10$, the sampling rate ($1/T_s$) should be

$$(1/T_s \geq 10/R \cong 10 \cdot B/0.5 = 20 \cdot B)$$

  at least 20 times the bandwidth B.
- The Nyquist-Shannon sampling theorem says that any time-continuous signal of bandwidth B can be reproduced faithfully from its sampled values if the sampling rate is at least $2 \cdot B$.
- *The recommended sampling rate for simple controllers is much higher than the theoretical minimum set by Shannon's theorem (the theorem makes no assumptions about implementation)*.

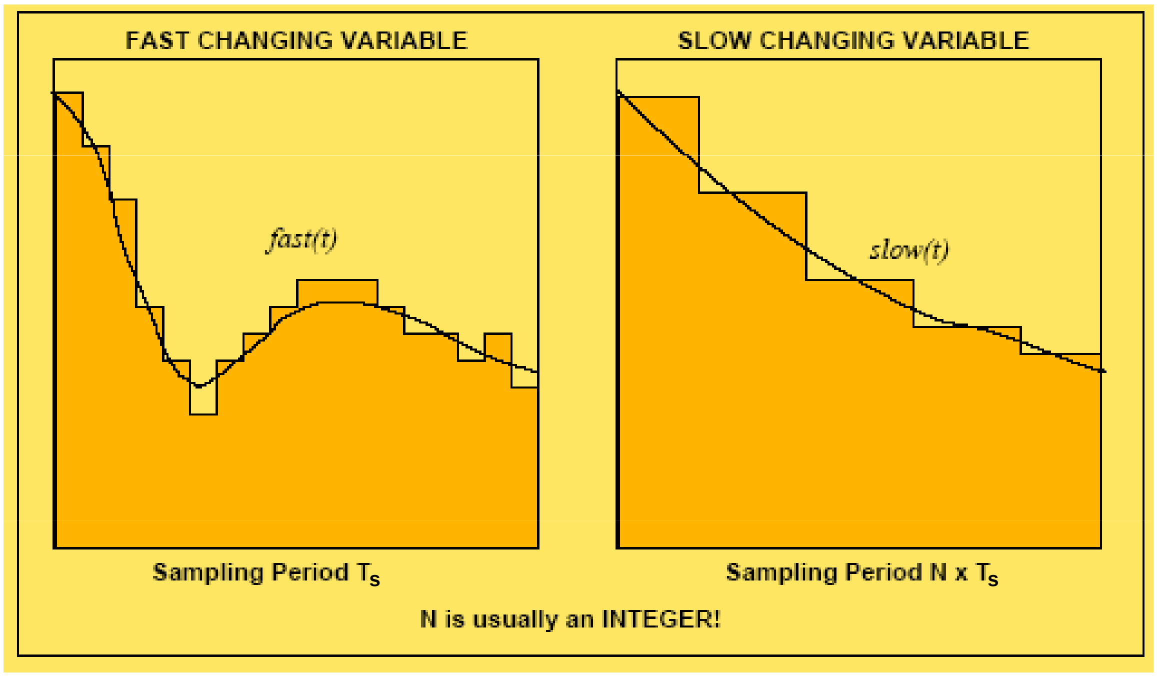
Multirate systems

- A system typically has state defined by multiple state variables (multivariate) e.g. rotation speed, temperature, fuel consumption, etc. of an engine.
- The state is monitored by multiple sensors and controlled by multiple actuators.
- Different state variables have different dynamics and will require different sampling periods to achieve smooth response. e.g. the rotation speed will change faster than its temperature.
- A system with multiple sampling rates is called a multirate system
- Multirate systems are very common since the technique is efficient in terms of computational effort but also allows the use of slower and thus cheaper sensors where appropriate.

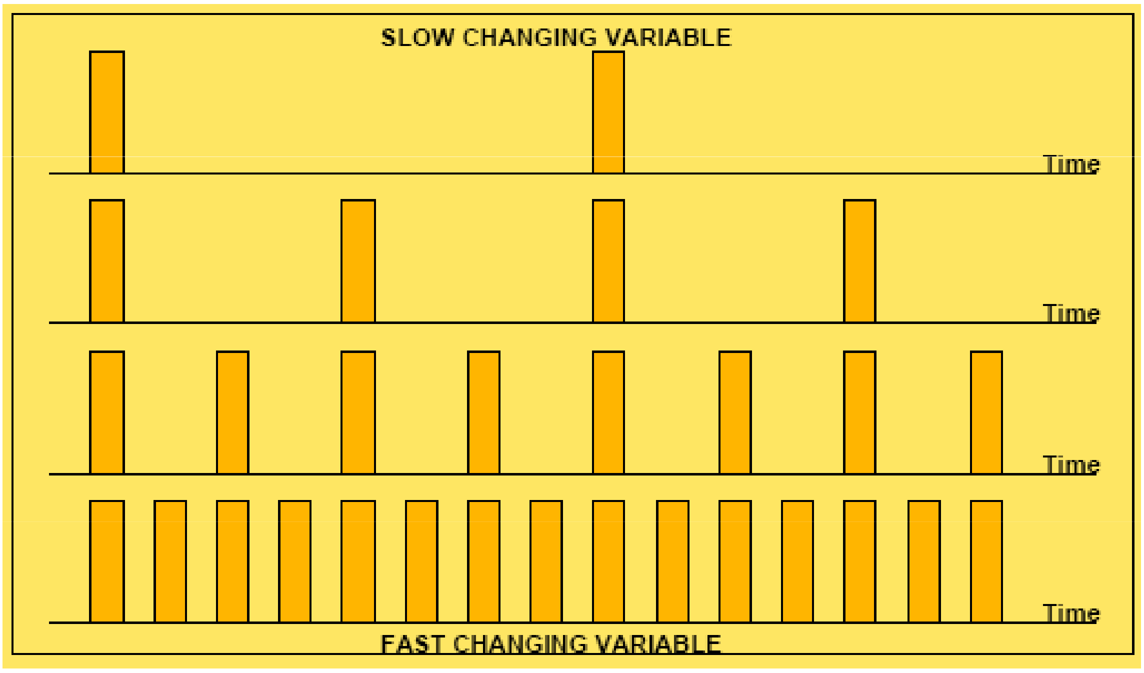
Sampling in multirate systems

- In multirate systems it is often useful to have the sampling rates related in a harmonic way so that longer sampling periods are integer multiples of shorter ones.
- This is useful because the state variables are usually not independent, and the relationships between them can be modeled better if longer sampling periods coincide with the beginning of the shorter ones.
- It is also frequently easier to design the system to sample at harmonically related intervals.

Sampling in multirate systems (2)



Sampling in multirate systems (3)



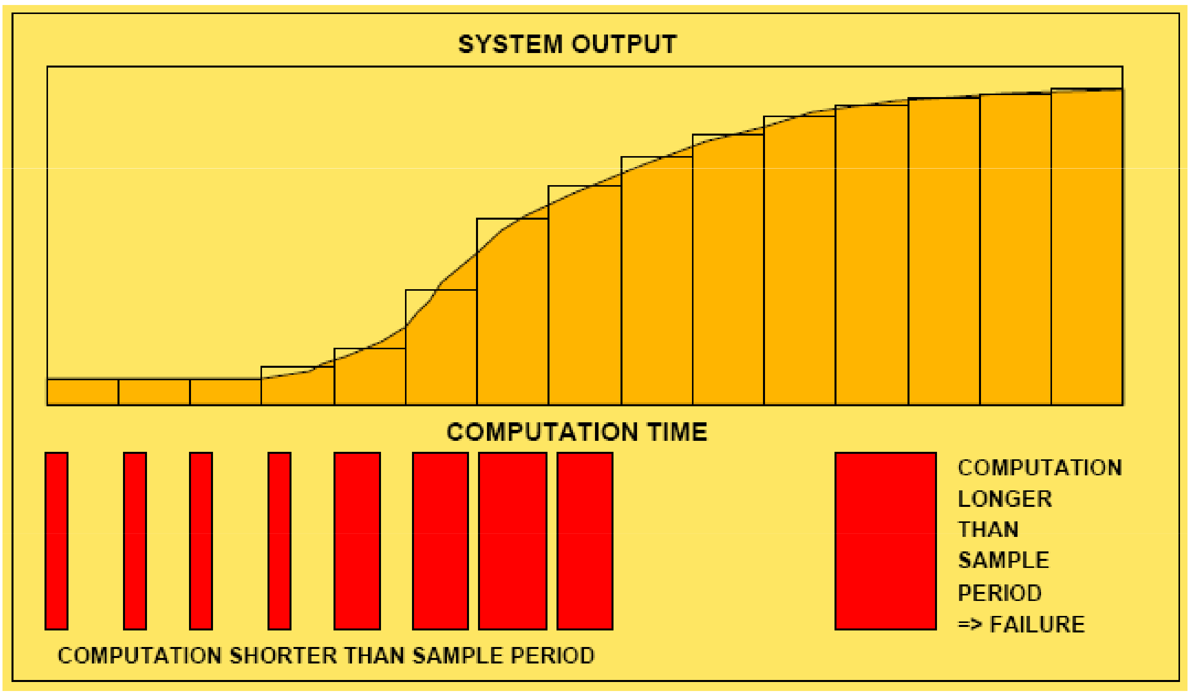
Timing Characteristics

- The workload generated by each multivariate, multirate digital controller consists of a few periodic control-law computations.
- A control system may contain many digital controllers, each dealing with part of the system.
- Together they demand perhaps hundreds of control-laws be computed: some periodically, others in reaction to some events.
- The control laws of each multirate controller may have harmonic periods and typically use the date produced by each other as inputs.

Control Law Computation Timing

- Each control-law computation can begin shortly after sampling (periodic task with period = T_s).
- Usually you want the computation complete, hence the sensor data processed, before the next sensor data is sampled (relative deadline $D \leq T_s$).
- This objective is met when the response time of each computation never exceeds the sampling period: ($f - r \leq T_s$).
- Situations where the control law takes longer to compute than the sampling period are highly undesirable as they usually result in the system failing.
- In a well designed system the aim is to not have unreasonable amounts of idle time but to always have enough time to complete the control law computation.

Control Law Computation Timing (2)


SYSTEM OUTPUT
COMPUTATION TIME
COMPUTATION SHORTER THAN SAMPLE PERIOD
COMPUTATION LONGER THAN SAMPLE PERIOD => FAILURE

Jitter

- In some cases the response time of the computation can vary from period to period.
- In some systems it is necessary to keep this variation small so that digital control outputs are available at instants more regularly spaced in time.
- In such cases we may impose a timing jitter requirement on the control-law computation. The variation in response time (jitter) does not exceed some threshold.

Assumptions for simplicity

- The simplicity of our digital controller depends on three assumptions:
    1. Sensors give accurate estimates of the state-variable values being monitored and controlled. This is not always true given noise or other factors.
    2. Sensor data give the state of the system. In general sensors monitor some observable attributes and the values of state variables have to be computed from the measured values.
    3. All parameters representing the dynamics of the system are known.

More complex computations

- set timer to interrupt periodically with period T;

```
at each clock interrupt do
    – Sample and digitize sensor readings;
    – Compute control output from measured values and
      state-variable values;
    – Convert control output to analogue form;
    – Estimate and update system parameters;
    – Compute and update state variables;
end do;
```

- The last 2 steps in the loop increase processing time.

High level control

- Controllers in complex systems are typically organized hierarchically.
- One or more digital controllers at the lowest level directly control the physical system.
- Each output of a higher-level controller is an input of one or more lower-level controllers.
- Usually one or more of the higher-level controllers interfaces with the operator.

Examples of Control Hierarchy

- A patient care system in a hospital:
  - Low-level controllers handling blood pressure, heart rate, respiration, glucose, etc.
  - High-level controller, e.g. an expert system which interacts with the doctor to choose desired values for the low-level controllers to maintain.
- This is a ‘two tier’ system.

Example of Control Hierarchy: ATC

Air Traffic Controller – Flight Management – Flight Control

- The hierarchy of flight control system, flight management avionics and air-traffic control systems.
- The air-traffic control system is at the highest level: it sets metering fixes and associated arrival times.
- The flight management system chooses the flight paths to reach the next metering fix at the requested time: it sets cruise speed, turn radius, descent rate, climb rate, heading, ... (parameters for the lower level controllers) accordingly.
- The flight controller at the lowest level handles pitch, roll and yaw inputs to achieve commanded speed, turn, climb/descent or other flight paths.

Example of Control Hierarchy: ATC (2)

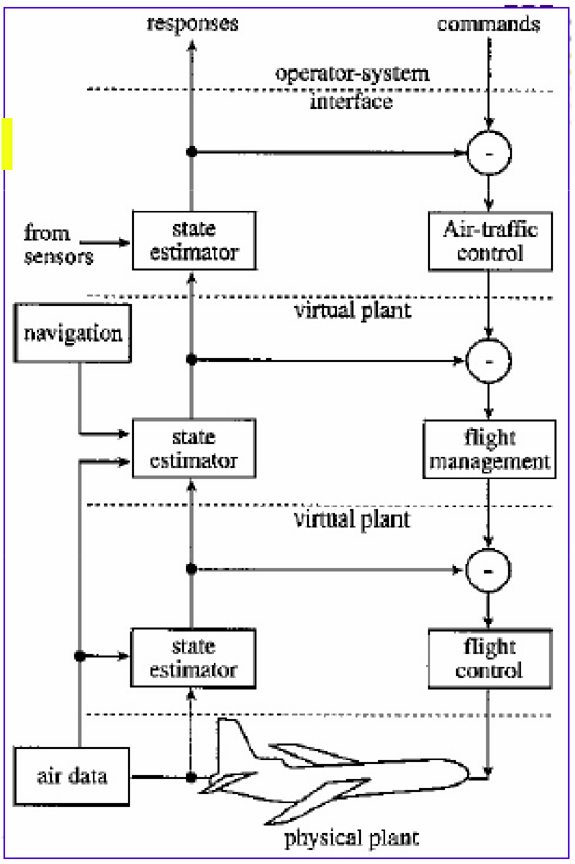
Air Traffic Controller – Flight Management – Flight Control

The Air Traffic Control	On-Board flight management	Flight controller
Regulates the flow of flight to each destination	**Chooses a time-referenced flight path**	**Receives sensor inputs and performs control**
Arrival time Metering Fix	**Arrival time Metering Fix**	**Plane control**

Example of Control Hierarchy: ATC (3)

High level controls

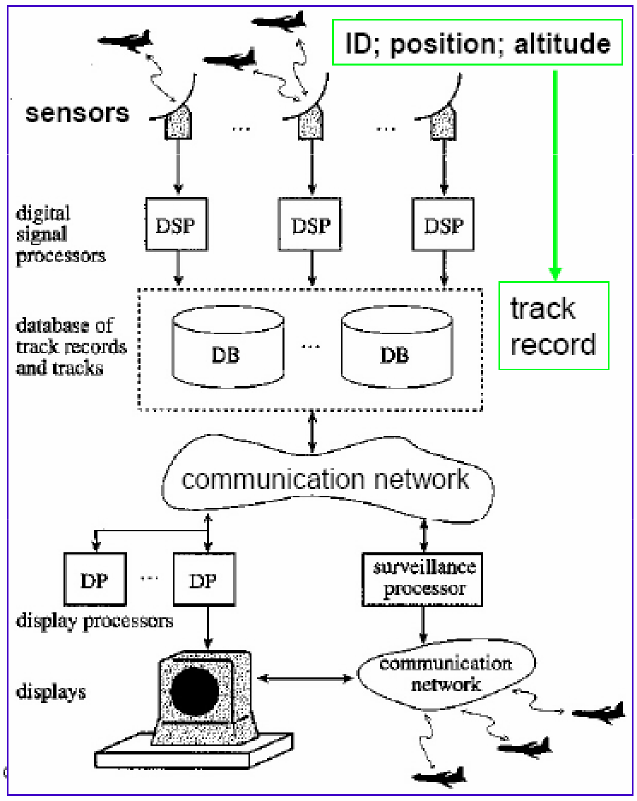
- While a digital controller deals with some dynamical behavior of the plant a second-level controller typically performs guidance and path planning functions:
  - most desirable trajectory among all trajectories that meet the constrains of the system.



Real-Time Command and Control

ATC - Air Traffic Controller

- The controller at the highest level of control hierarchy is a Command and Control System (CCS)
  - timing requirements of a CCS system are less stringent



R-T applications: signal processing

Signal processing

- Most signal processing applications are real-time, requiring response times from less than a millisecond up to seconds; e.g. digital filtering, video-audio compression/decompression, and radar signal processing.
- Typically a real-time signal processing application computes in one sampling period, one or more outputs, each being a weighted sum of *n* inputs (weights may even vary over time): the processing time required to produce one output is *O(n)*.

R-T applications: signal processing (2)

Signal processing

- The processing time demand of an application depends on the sampling period and how many outputs are required to be produced per sampling period: if the number of outputs is also of the order *n*, then the complexity of the computation is $O(n^2)$.
- For digital filtering of audio signals, the sampling rate can be tens of kHz (44.1 kHz for CD digital audio) and hundreds of terms may have to be computed, hence tens of millions of multiplications and additions per second may be required.
- Real-time video compression ($n \times n$ pixels, 30 frames/s) may have complexity of order n^4: with $n = 100$, $30 \times (100)^4 = 3 \cdot 10^9$ multiplications and additions per second are required!
  - by dividing each image into squares of 8×8 pixels, the number is reduced to $30 \times 8^2 \times 8^2 \times (n/8) \times (n/8) = 30 \times (8 \times n)^2$
  - with $n = 100$ we reduce it to $1.92 \cdot 10^7$

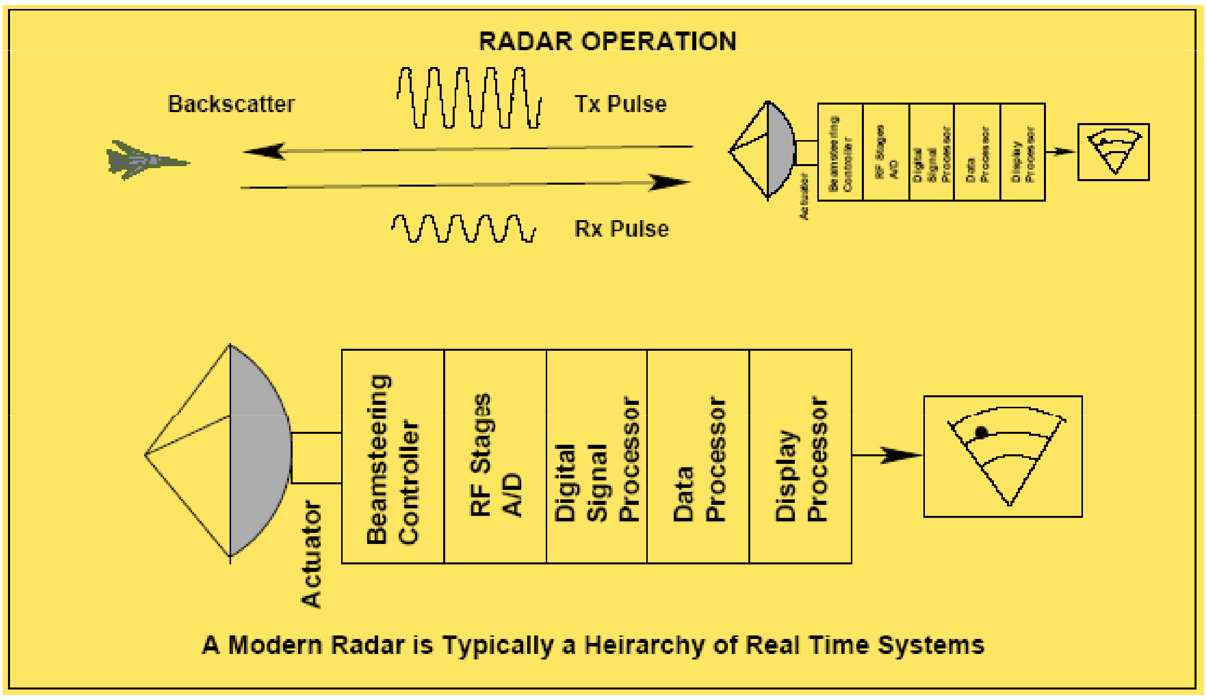
R-T applications: signal processing (3)

Signal processing: radar system

- Signal processing is usually part of a larger system. e.g. a passive radar signal processing system.
- The system comprises an I/O subsystem that samples and digitizes the echo signal from the radar and places the sampled values in shared memory.
- An array of digital signal processors process these values.
- The data produced are analyzed by one or more data processors which not only interface to the display system but also feed back commands to control the radar and select parameters for the signal processors to be used for the next sampling period.

R-T applications: signal processing (4)

Signal processing: radar system

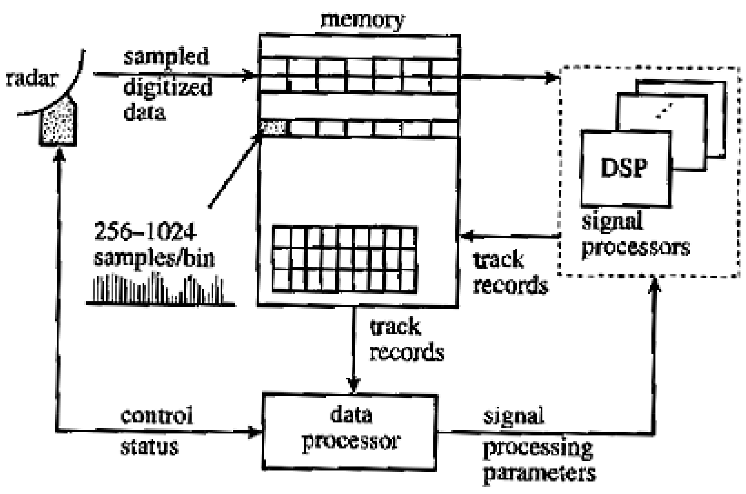


R-T applications: signal processing (5)

Signal processing: radar system

- Real-time signal processing application computes in each sampling period one or more outputs
- Each output is a weighted sum of n inputs *y(i)*'s

$$x(k) = \sum_{i=1}^{n} a(k,i)y(i)$$



R-T applications: R-T databases

Real-Time databases

- Examples:
  - Stock exchange price database systems.
  - Air Traffic Control databases.
  - …
- What makes them real-time?
  - The data is ‘perishable’.
  - The data values are updated periodically.
  - After a while the data has reduced value.
  - There needs to be temporal consistency as well as normal data consistency.

R-T applications: R-T databases (2)

Real-Time databases: absolute temporal consistency

- Real-time data has parameters such as age (the time since the data was captured) and temporal dispersion.
- The age of an object measures how up-to-date the information is.
- The age of an object whose value is computed from other objects is equal to that of the oldest of those objects.
- A set of data objects is said to be absolutely temporally consistent if the maximum age in the set is no greater than a certain threshold.

R-T applications: R-T databases (3)

Real-Time databases: relative temporal consistency

- A set of data objects is said to be relatively temporally consistent if the maximum difference in the ages is less than the relative consistency threshold used by the application.
- For some applications the absolute age is less important than the differences in ages.

R-T applications: R-T databases (4)

Real-Time databases: consistency models

- Concurrency control mechanisms such as 2-phase locking (prepare / commit-rollback) have been used to ensure serializability of read and update transactions and maintain data integrity of non-real-time databases.
- These data-consistency mechanisms can make it more difficult for updates to be completed in time: late updates may cause data to become temporally inconsistent.

R-T applications: R-T databases (5)

Real-Time databases: consistency models

- Weaker data-consistency models are sometimes used to ensure the timeliness of updates and reads.
- For instance we may require updates to be serializable but allow read-only transactions not to be serializable.
- Usually the more relaxed the serialization requirement, the more flexibility the system has in interleaving the read and write operations from different transactions, and the easier it is to schedule transactions so that they complete in time.

Summary of Real-Time applications

1. Purely cyclic: Every task executes periodically. Even I/O operations are polled. Demands on resources do not vary significantly from period to period. Most digital controllers are of this type.
2. Mostly cyclic: Most tasks execute periodically. The system can also respond to some external events asynchronously.
3. Asynchronous and somewhat predictable: In applications such as multimedia communication, radar signal processing, tracking, most tasks are not periodic. Duration between executions of a task may vary considerably or the resource requirements may vary. Variations have bounded ranges or known statistics.
4. Asynchronous and unpredictable.