Introduction to Real-Time Systems

Minsoo Ryu

RTCC Lab. at Hanyang University

What is Real-Time Computing ?

❒ Misconceptions

- Real-time computing is equivalent to fast computing
- The objective of real-time computing is to minimize the response time of a given set of tasks

❒ Theoretical definition

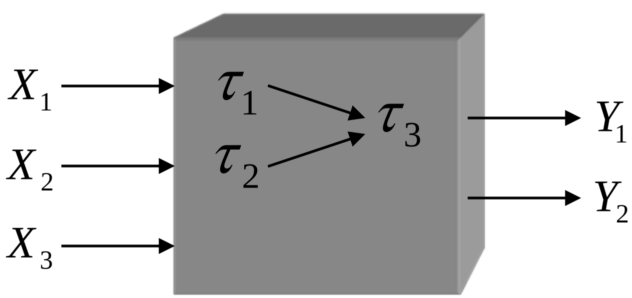
- The correctness of computing depends not only on the correctness of its logical result but also on the result delivery time
- In addition, real-time computing must be predictable

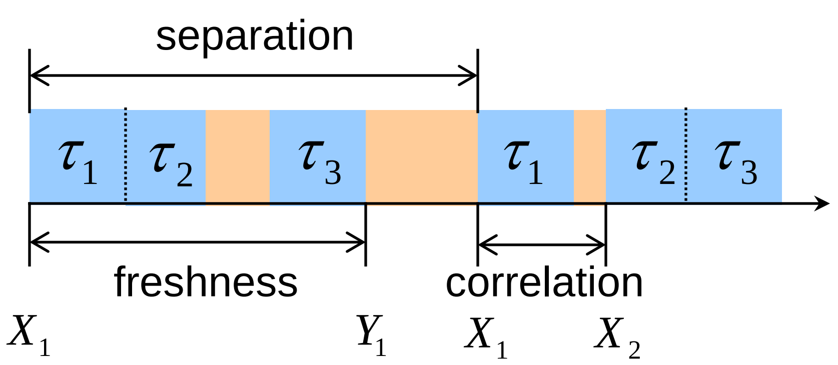
Classification of Timing Requirements

❒ Three types of timing requirements

- **Freshness -> deadline**
  - The time delay for data to flow through the system
- **Separation -> period**
  - The time interval between two consecutive activations (completions)
- **Correlation -> synchronization**
  - The time skew between several inputs to produce an output

왜곡하다

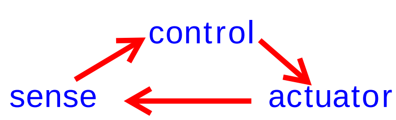




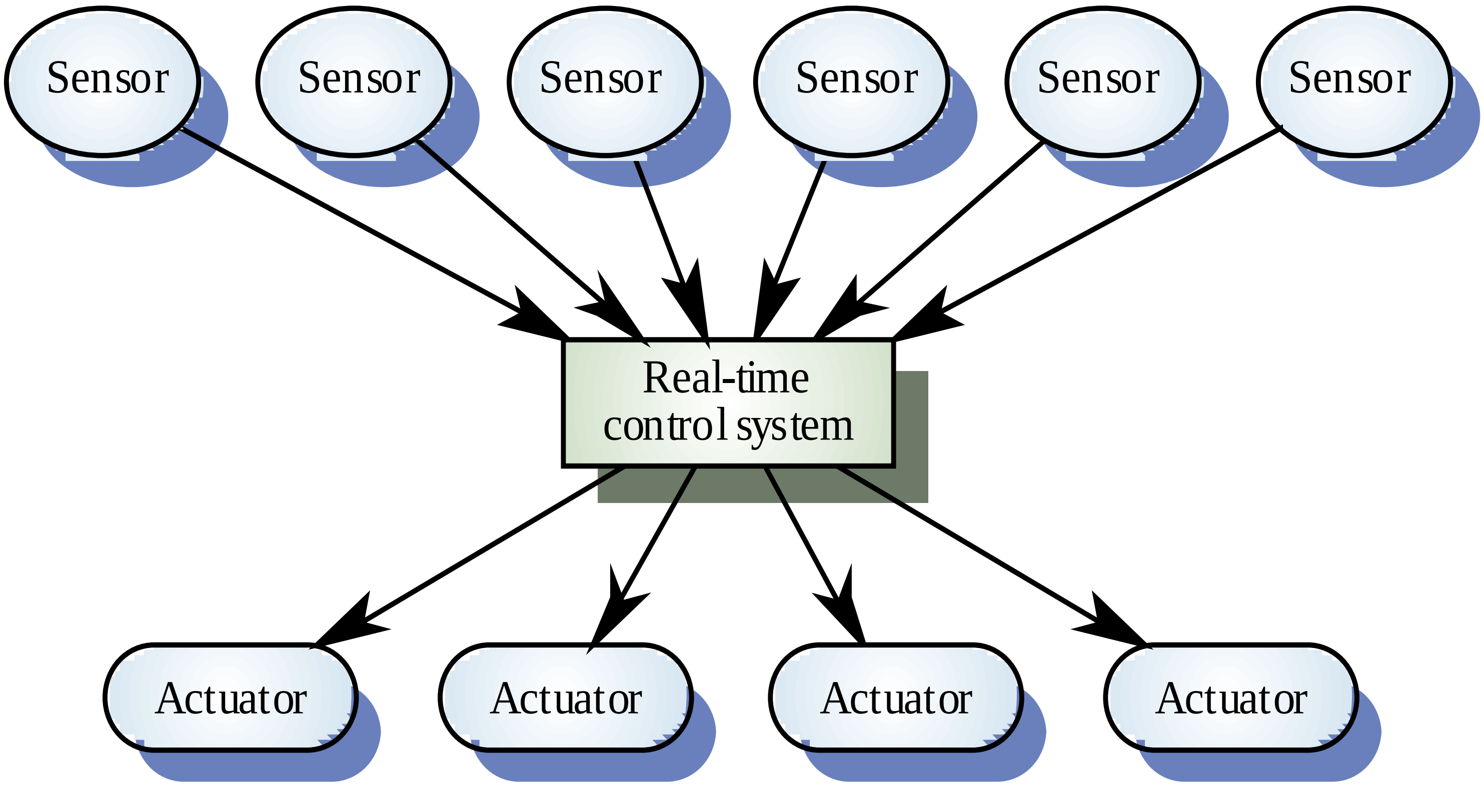
Typical Real-Time Systems

- **Automatic control systems** -> deadline,period 존재

- **Such systems monitor and control their environment**
- **Inevitably associated with hardware devices**
  - **Sensors: Collect data from the system environment**
  - **Actuators: Change (in some way) the system's environment**
- **Time is critical**
  - **Real-time systems MUST respond within specified times**

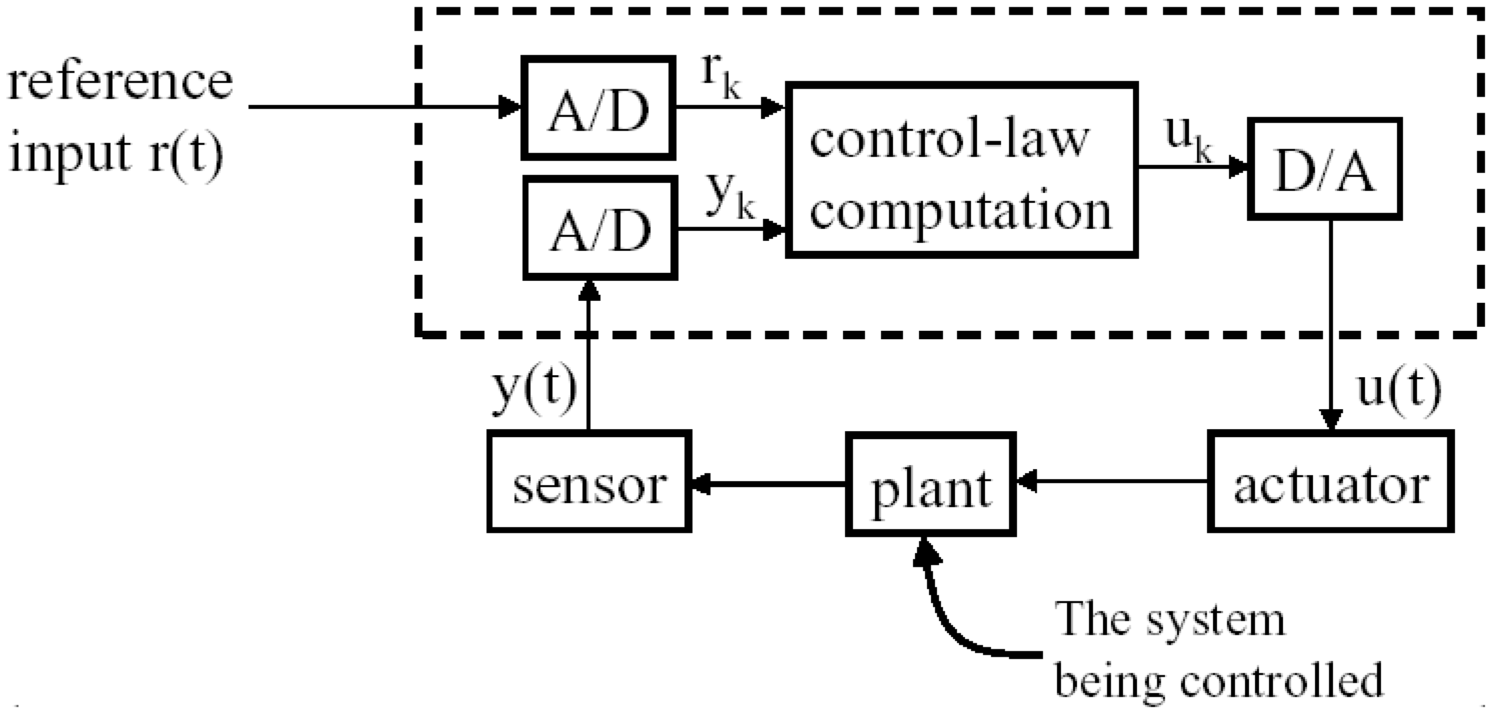


Real-Time Control System Structure



A Simple RT Control System Model

❒ **SISO (Single Input Single Output) control loop**



Control Loop Implementation

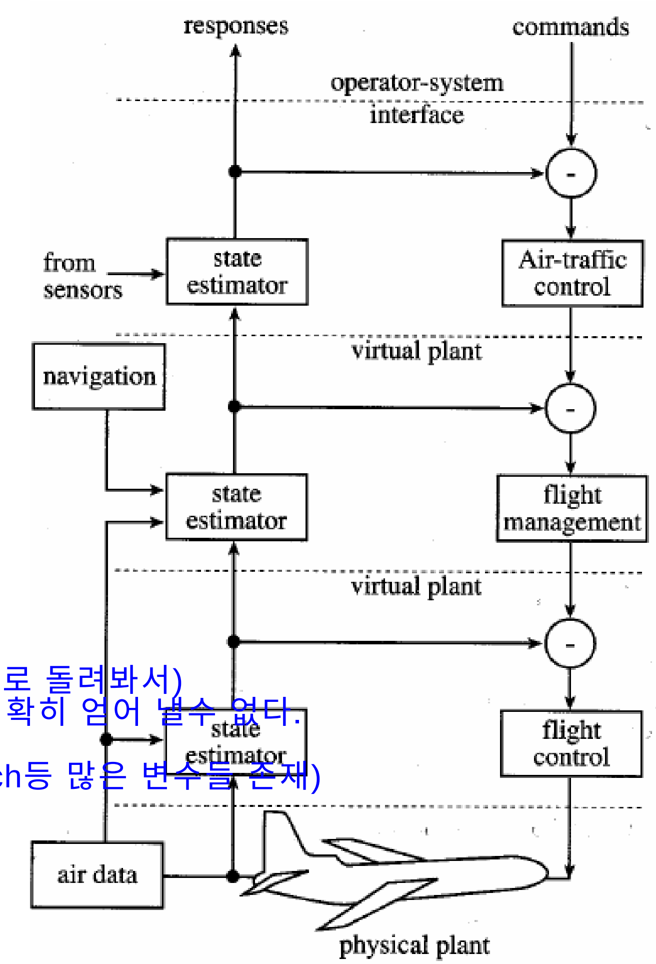
- **Pseudo code for the SISO control system**

```
set timer to interrupt periodically with period T;
at each timer interrupt do
    do analog-to-digital conversion to get y;
    compute control output u;
    output u and do digital-to-analog conversion;
od
```

  - **T (sampling period)**
    - Design choice between a lower bound and an upper bound
- **Timing requirements**
  - **Control systems have periodicity requirements, and therefore deadline requirements to complete periodic jobs**

Other Applications

- ❒ **Air traffic and flight control**
  - **Hierarchy model**



- ❒ **Other applications include**
  - **Radar surveillance system**
    감시
  - **Robot control system**
  - **Cruise control system**

일반적인 프로그램 개발 순서
1. requirement
real time의 경우
time req 추가

2. design
behavior
(real time scheduling)

3. implementation
time analysis
1.코드의 실행 시간 예측 (실제로 돌려봐서)
측정시 -> worst case 고려 정확히 얻어 낼수 없다.
2. binary로 실제 시간 예측
(하지만 , cache miss & branch등 많은 변수들 존재)
따라서 예측 값을 사용

4. v&v(verification&validation)
time requirement 검증

Hard and Soft Real-Time Systems

- ❒ **Hard deadline**
  - A deadline miss results in a catastrophe
  - Probabilistic perspective: deadline miss probability is zero
- ❒ **Soft deadline**
  - Deadline misses are allowed, but degrades system performance
  - Probabilistic perspective: deadline miss probability is small
- ❒ **Firm deadline**
  - Completing a task after its deadline is not useful and may even be harmful

Hard and Soft Real-Time Systems

❐ Guaranteed service

- **The user wants guarantees on services**
- **Hard real-time or soft real-time guarantees**
- **Hard real-time applications**
  - Control systems
  - Database systems
- **Soft real-time applications**
  - Multimedia and network applications with service guarantees

❐ Best-effort service

- **The system attempts to provide best services with no guarantees**

RT Scheduling: RM (Rate Monotonic)

❐ Assumptions

- Processes have periods, worst-case execution times (WCETs), and deadlines

❐ Scheduling policy

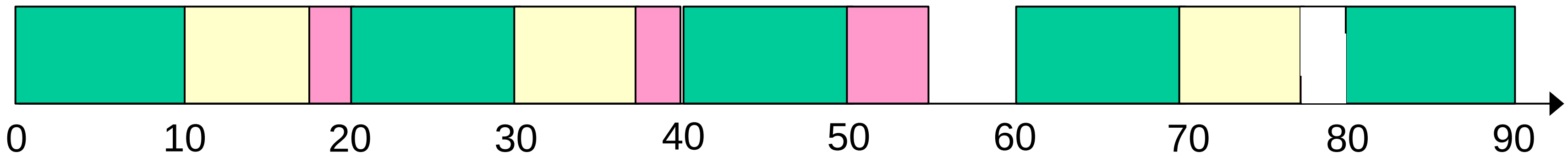
- Give higher priorities to tasks with shorter periods
- Preemptive static priority scheduling

❐ Optimality

- If a feasible static priority assignment exists for some process set, the RM priority assignment is feasible for that process set

RT Scheduling: RM (Rate Monotonic)

- Consider the following tasks
  - Process X: period = 20, WCET = 10, deadline = 20
  - Process Y: period = 30, WCET = 8, deadline = 30
  - Process Z: period = 40, WCET = 4, deadline = 40



- Schedulability test
  - CPU utilization: $U = \sum_{i=1}^{m} e_i / p_i$
  - A set of m processes is schedulable if $U \le m(2^{1/m} - 1)$
    - For large m, $m(2^{1/m} - 1) \approx \ln 2 \approx 0.69$

RT Scheduling: EDF (Earliest Deadline First)

❒ Scheduling policy

- Give higher priorities to tasks with earlier absolute deadlines
- Preemptive dynamic priority scheduling

❒ Optimality

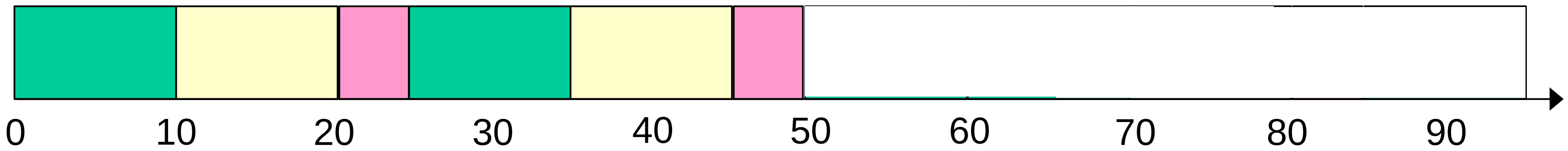
- If a feasible dynamic priority assignment exists for some process set, the EDF priority assignment is feasible for that process set

❒ Schedulability test

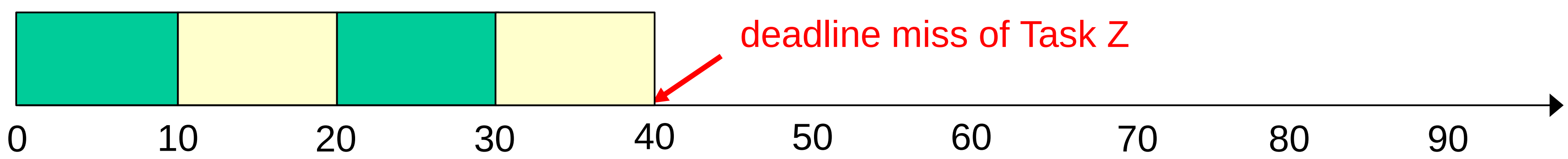
- A set of m processes is schedulable if and only if $U \leq 1$

RT Scheduling: EDF (Earliest Deadline First)

- Consider the following tasks
  - Process X: period = 20, WCET = 10, deadline = 20
  - Process Y: period = 30, WCET = 10, deadline = 30
  - Process Z: period = 40, WCET = 5, deadline = 40



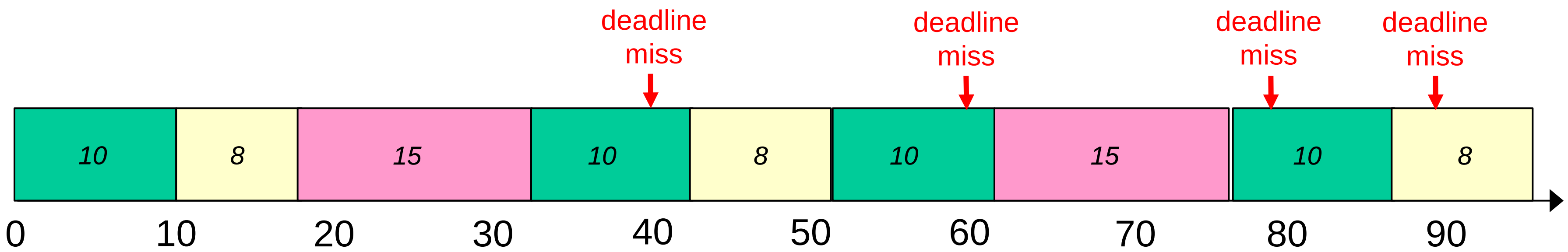
EDF schedule



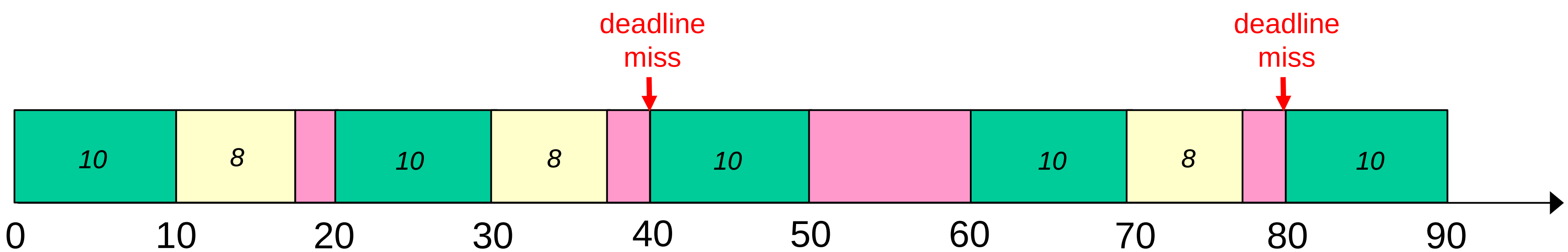
RM schedule

Non-schedulable Behavior

- Consider the following tasks
  - Process X: period = 20, WCET = 10, deadline = 20
  - Process Y: period = 30, WCET = 8, deadline = 30
  - Process Z: period = 40, WCET = 15, deadline = 40



EDF schedule



RM schedule