Real Time Systems

Juan José Costa and Alejandro Pajuelo

Escola Tècnica Superior de Telecomunicacions de Barcelona (ETSETB)

Universitat Politècnica de Catalunya (UPC)

2021-2022Q2

Index

- Why Real Time Systems?
- What are Real Time Systems?
- RTS types
- RTS architecture
- RTS programming
- Examples

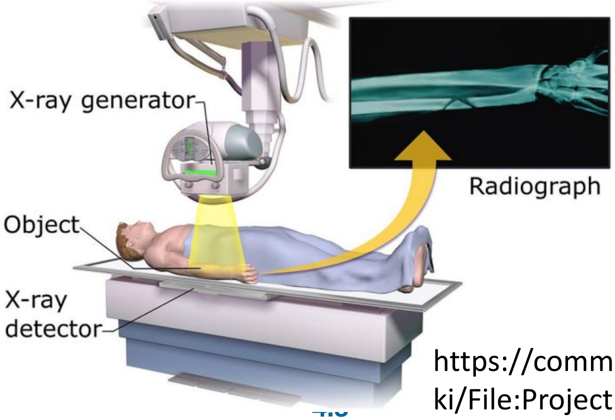
Why Real Time Systems?

Why Real Time Systems?

https://commons.wikimedia.org/wiki/File:LGwashingmachine.jpg

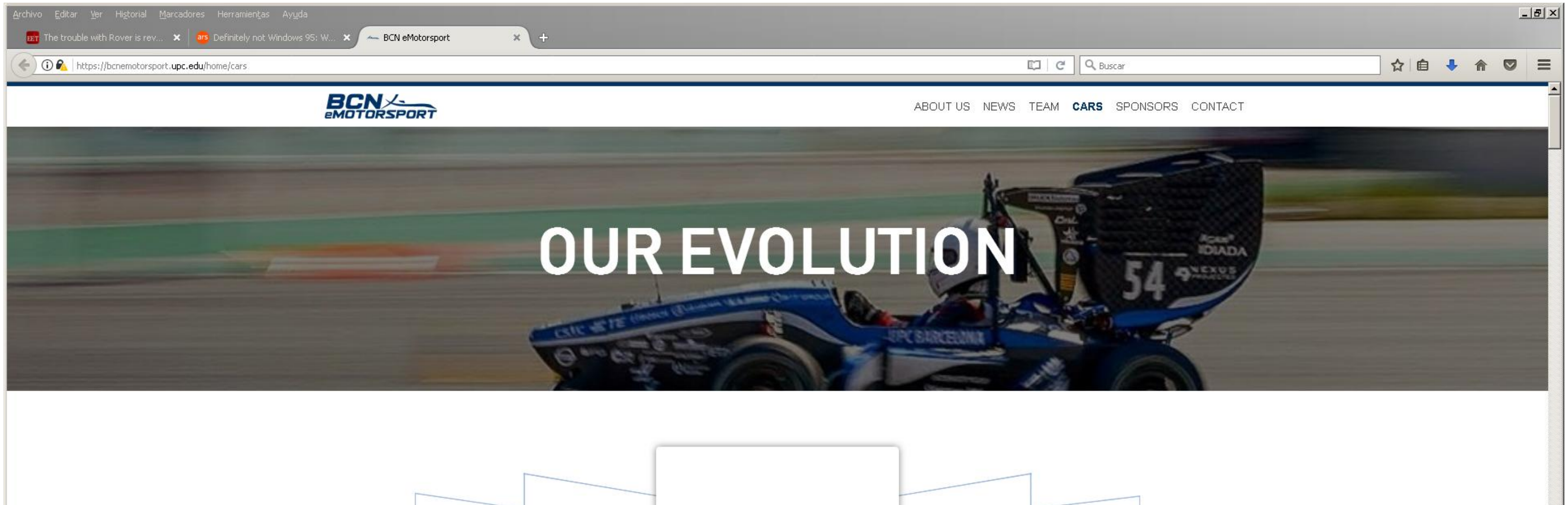
Why Real Time Systems?

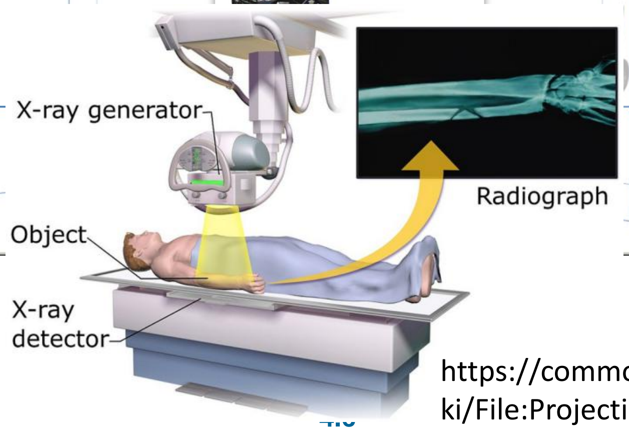
https://commons.wikimedia.org/wiki/File:LGwashingmachine.jpg



https://commons.wikimedia.org/wiki/File:Projectional_radiography_components.jpg

Why Real Time Systems?


Archivo Editar Ver Historial Marcadores Herramientas Ayuda
The trouble with Rover is rev...
Definitely not Windows 95: W...
BCN eMotorsport
https://bcnemotorsport.upc.edu/home/cars
Buscar
BCN eMOTORSPORT
ABOUT US
NEWS
TEAM
CARS
SPONSORS
CONTACT
OUR EVOLUTION
CAT13e
CAT11e

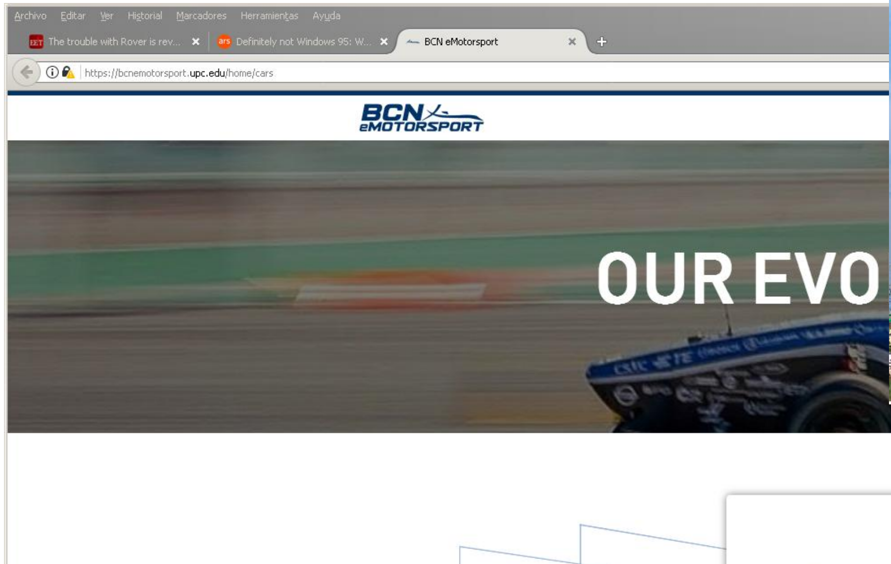


https://bcnemotorsport.upc.edu

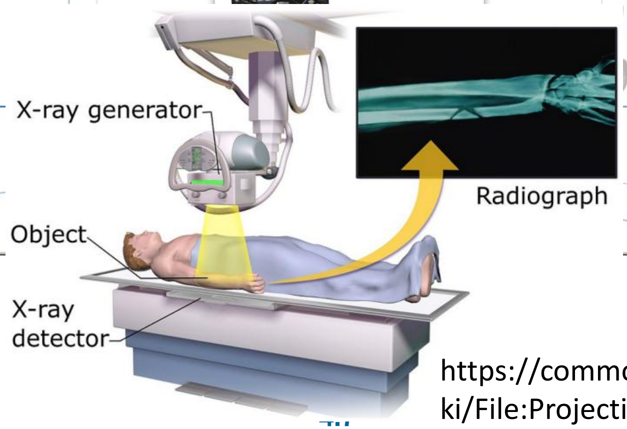
https://commons.wikimedia.org/wiki/File:LGwashingmachine.jpg

https://commons.wikimedia.org/wiki/File:Projectional_radiography_components.jpg

Why Real Time Systems?


Archivo Editar Ver Historial Marcadores Herramientas Ayuda
The trouble with Rover is rev...
Definitely not Windows 95: W...
BCN eMotorsport
https://bcnemotorsport.upc.edu/home/cars
BCN eMOTORSPORT
OUR EVO
CAT13e
CAT11e

https://commons.wikimedia.org/wiki/File:Kernkraftwerk_Grafenrheinfeld_-_2013.jpg



https://bcnemotorsport.upc.edu

https://commons.wikimedia.org/wiki/File:LGwashingmachine.jpg

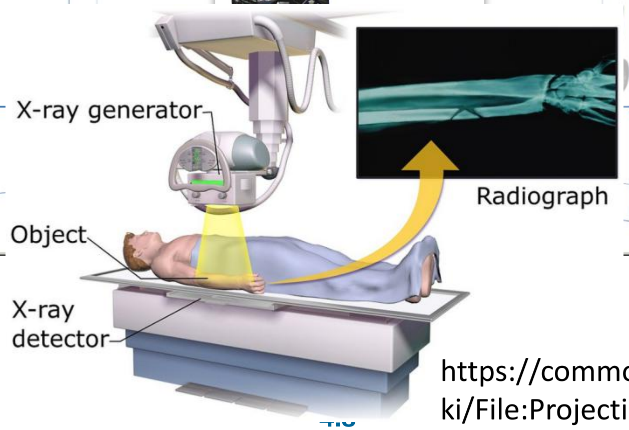
https://commons.wikimedia.org/wiki/File:Projectional_radiography_components.jpg

Why Real Time Systems?

[Krywko20]

https://commons.wikimedia.org/wiki/File:Kernkraftwerk_Grafenrheinfeld_-_2013.jpg

https://commons.wikimedia.org/wiki/File:LGwashingmachine.jpg



https://commons.wikimedia.org/wiki/File:Projectional_radiography_components.jpg

https://bcnemotorsport.upc.edu

Index

- Why Real Time Systems?
- What are Real Time Systems?
- RTS types
- RTS architecture
- RTS programming
- Examples

What are Real Time Systems?

- Oxford dictionary:

  “Relating to a system in which input data is processed within milliseconds so that it is available virtually immediately as feedback to the process from which it is coming.”

What are Real Time Systems?

- Oxford dictionary:

  “Relating to a system in which input data is processed within milliseconds so that it is available virtually immediately as feedback to the process from which it is coming.”

- Oxford dictionary of Computing:

  “Any system in which the time at which output is produced is significant. This is usually because the input corresponds to some movement in the physical world, and the output has to relate to that same movement. The lag from input time to output time must be sufficiently small for acceptable timeliness.”

What are Real Time Systems?

- Oxford dictionary:

  “Relating to a system in which input data is processed within milliseconds so that it is available virtually immediately as feedback to the process from which it is coming.”

- Oxford dictionary of Computing:

  “Any system in which the time at which output is produced is significant. This is usually because the input corresponds to some movement in the physical world, and the output has to relate to that same movement. The lag from input time to output time must be sufficiently small for acceptable timeliness.”

- *response time* – Time to generate an output from an input

What are Real Time Systems?

“A ***real-time system*** *is any information processing system which has to respond to externally generated input stimuli within a finite and specified period”* [Burns09]

What are Real Time Systems?

*“A **real-time system** is any information processing system which has to respond to externally generated input stimuli within a finite and specified period”* [Burns09]

- Windows™ ?
  - – User enters commands
  - – Expect result in a few seconds

What are Real Time Systems?

*“A **real-time system** is any information processing system which has to respond to externally generated input stimuli within a finite and specified period”* [Burns09]

- Windows™ ?
  - User enters commands
  - Expect result in a few seconds
  - What happens if it lasts more?

Index

- Why Real Time Systems?
- What are Real Time Systems?
- RTS types
- RTS architecture
- RTS programming
- Examples

RTS types

- **Failure** is a distinctive factor for RTS
  - Correctness depends on
    - the logical result
    - **AND** the time it was delivered
  - Failure to respond at deadline →as bad as a wrong!
- Depending on the tolerance to response time:
  - Hard Real Time
  - Firm Real Time
  - Soft Real Time

Hard Real Time Systems

a) Deadline MUST be ensured

b) No late delivery allowed

Soft Real Time Systems

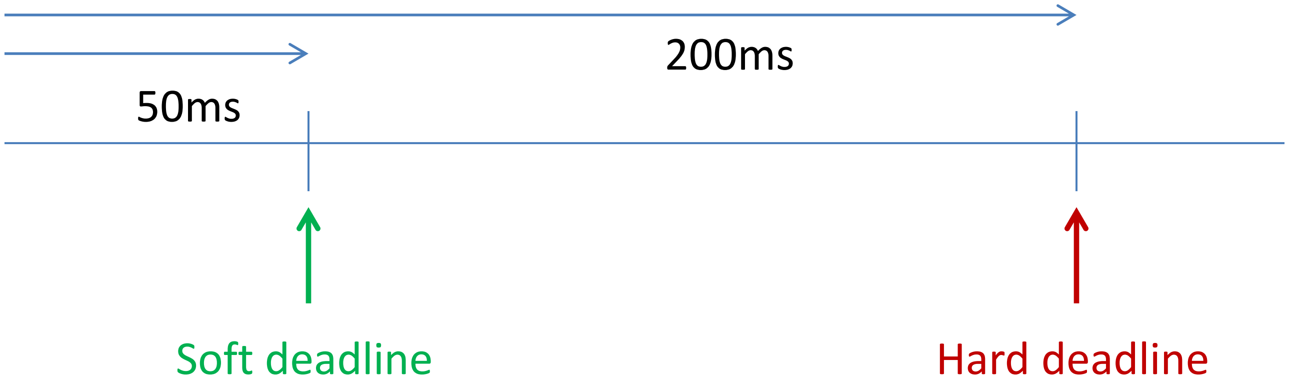
- Response times are important, but…

a) Deadlines can be **missed** occasionally

b) Service can ocassionally be **delivered late**

Example

- Systems may mix different types of deadline
- Ex: System with warning event:



Firm Real Time Systems

- Response time are important, but…

a) Deadlines can be missed occassionally

b) There is **no value in delivering a late result**

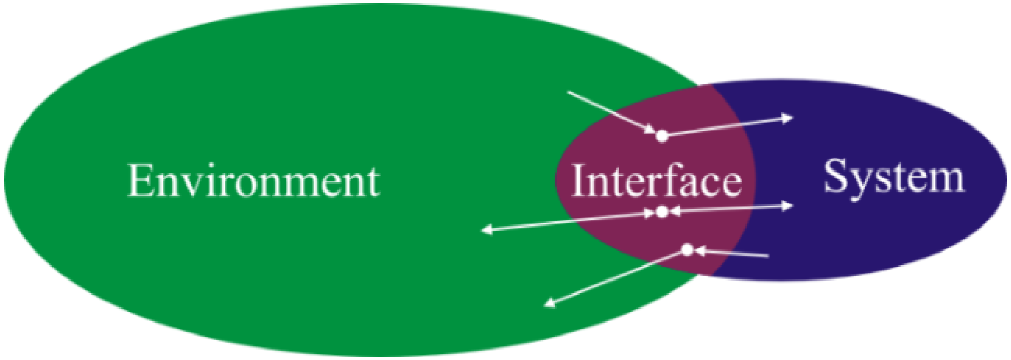
Everything clear?

Index

- Why Real Time Systems?
- What are Real Time Systems?
- RTS types
- RTS architecture
- RTS programming
- Examples

Embedded Computer Systems

- In RTS, a **computer** is interfaced directly to some **physical equipment** and it is **dedicated** to monitor or control its operation
  - (Different from general purpose systems)



[Harder18]

RTS elements

- Environment
- Real time hardware
- Real time software

Environment

- Needs to be modelled *(out of scope)*
- It can be measured (monitor)
- We can interactuate with it (control)

Simple real time system(1 component)

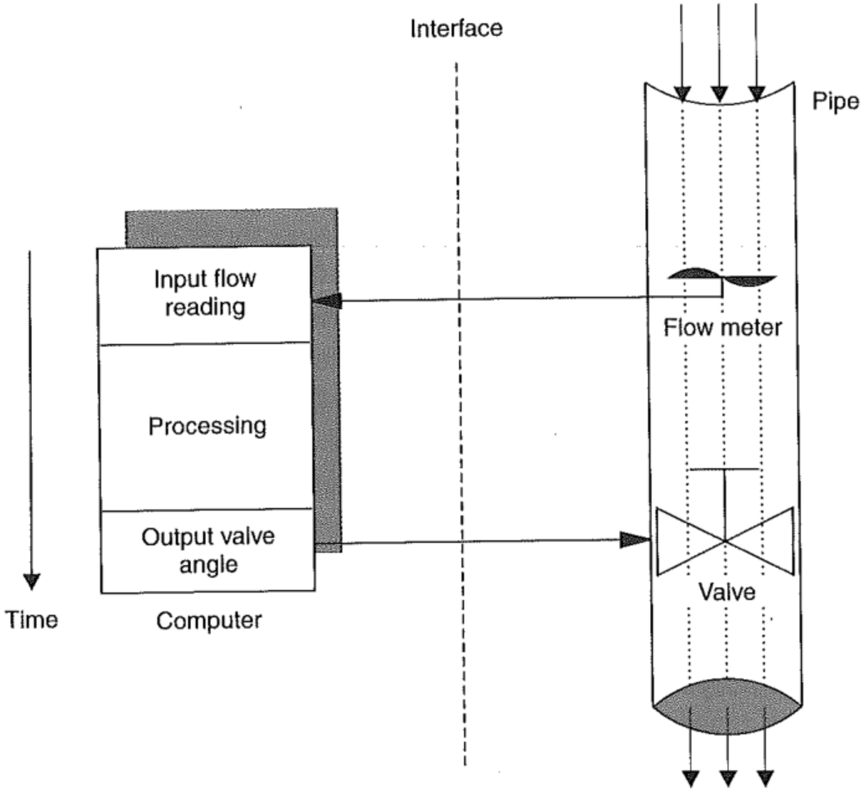


Figure 1.1 A fluid control system.

[Burns09]

Simple real time system(1 component)

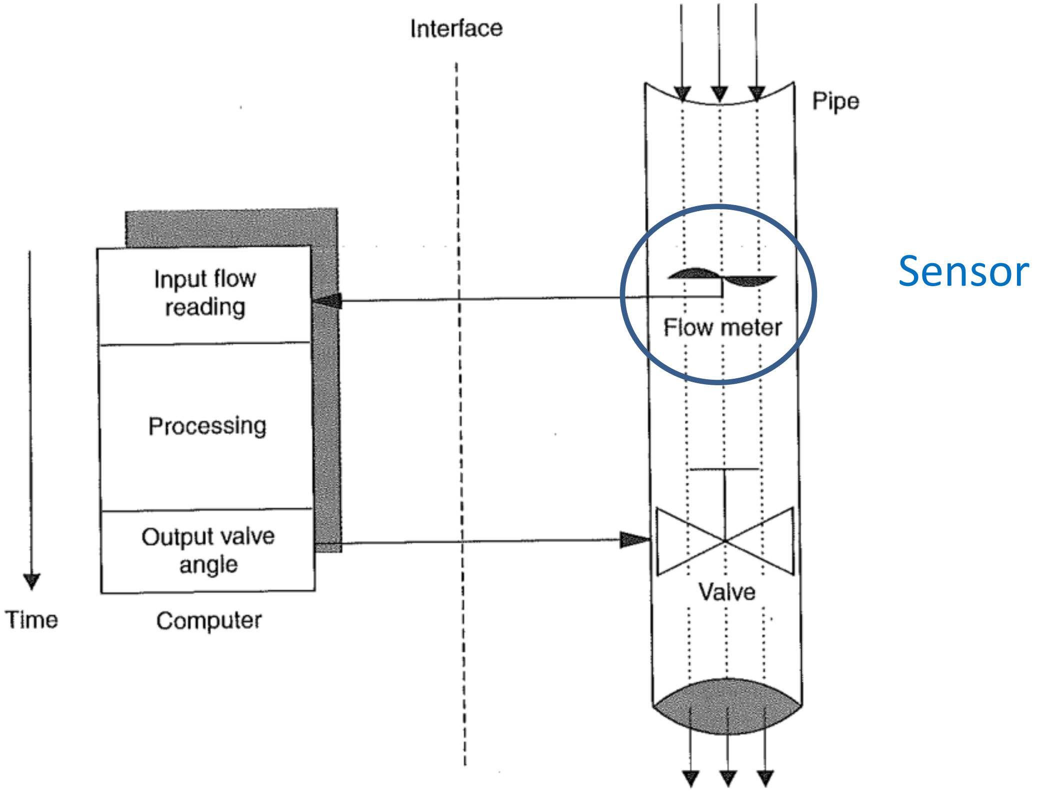
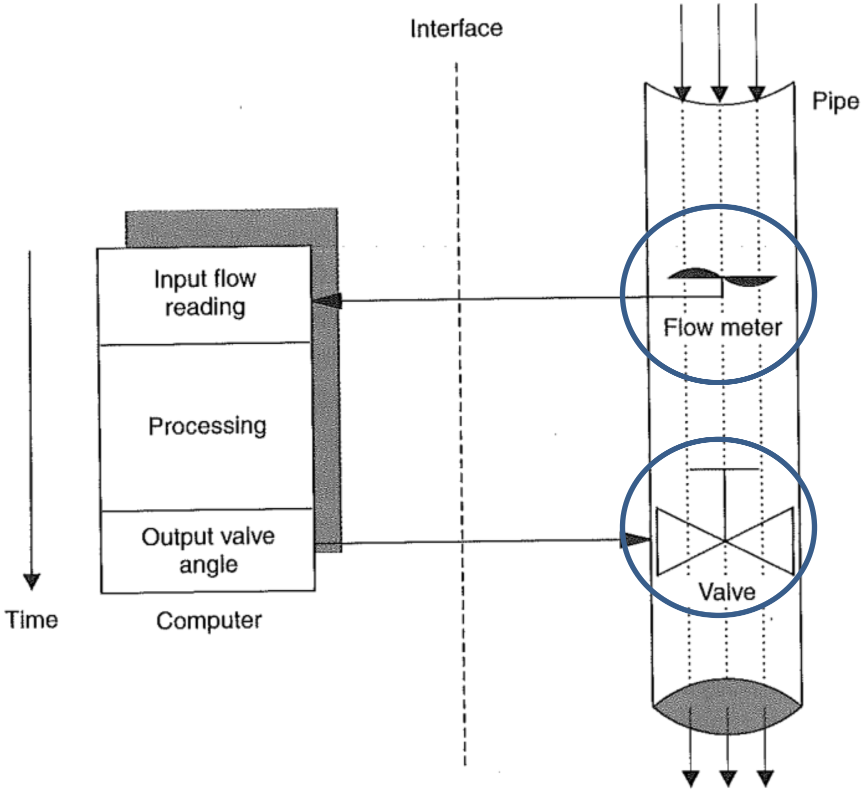


Figure 1.1 A fluid control system.

[Burns09]

Simple real time system(1 component)



Sensor

Actuator

Figure 1.1 A fluid control system.

[Burns09]

Simple real time system(1 component)

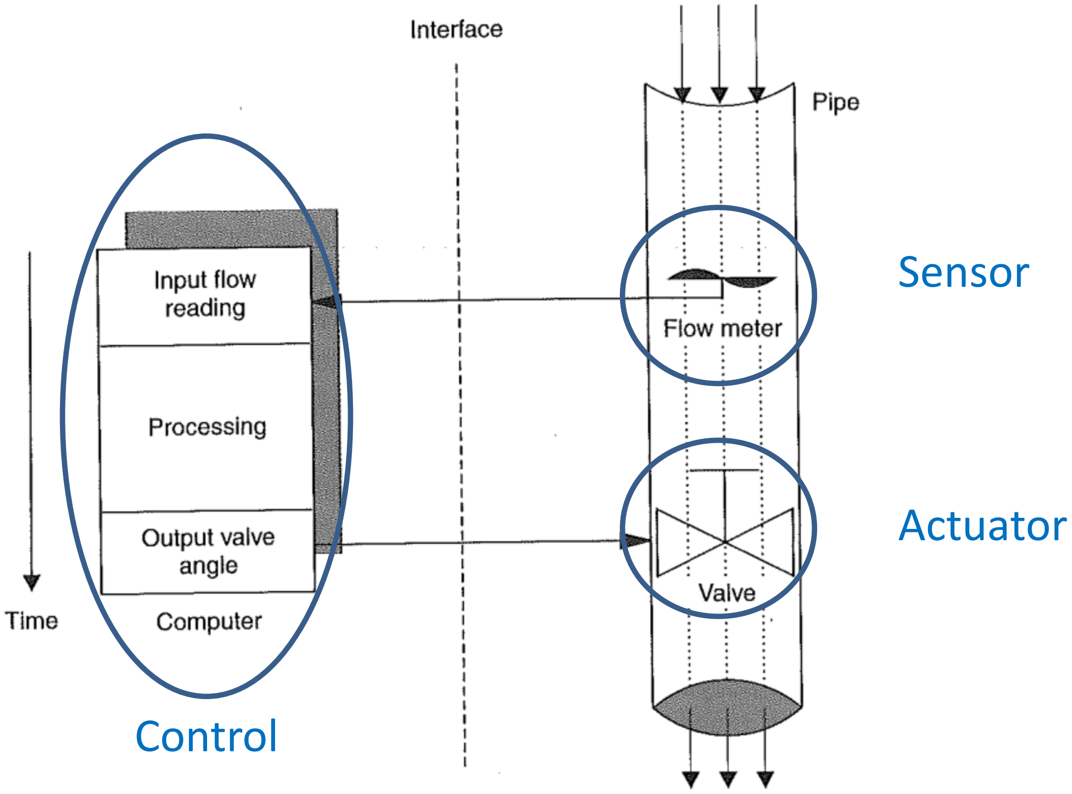


Figure 1.1 A fluid control system.

[Burns09]

Simple real time system(1 component)

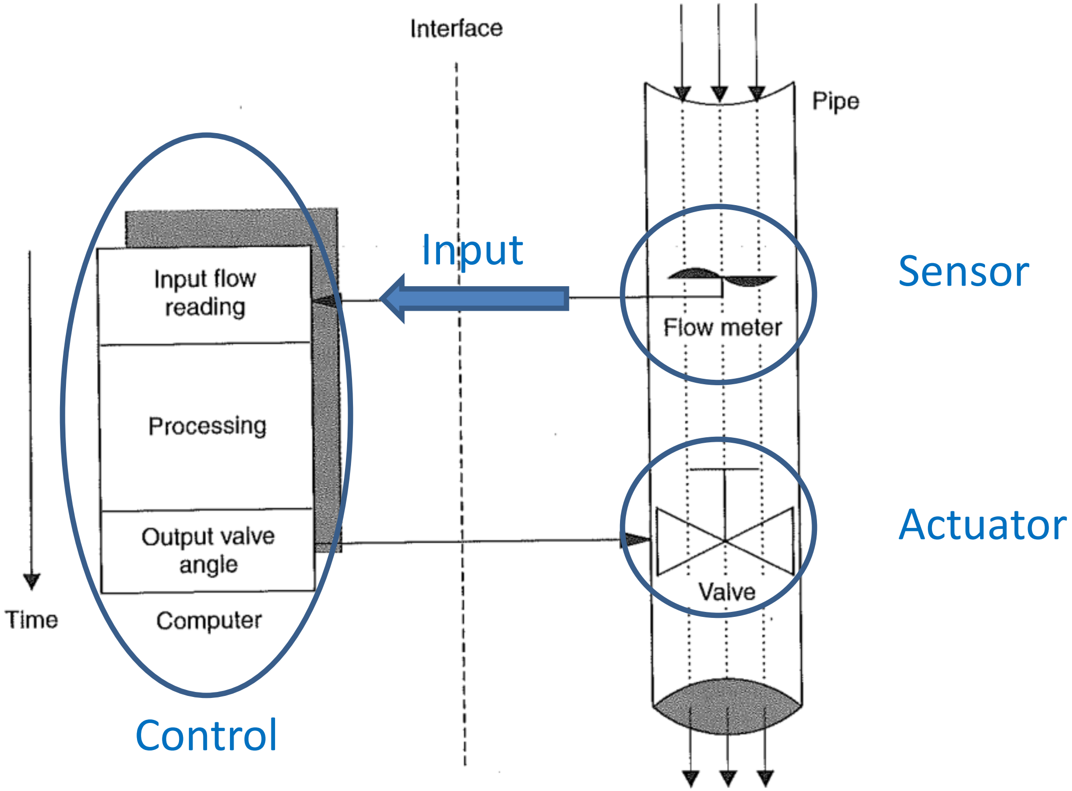
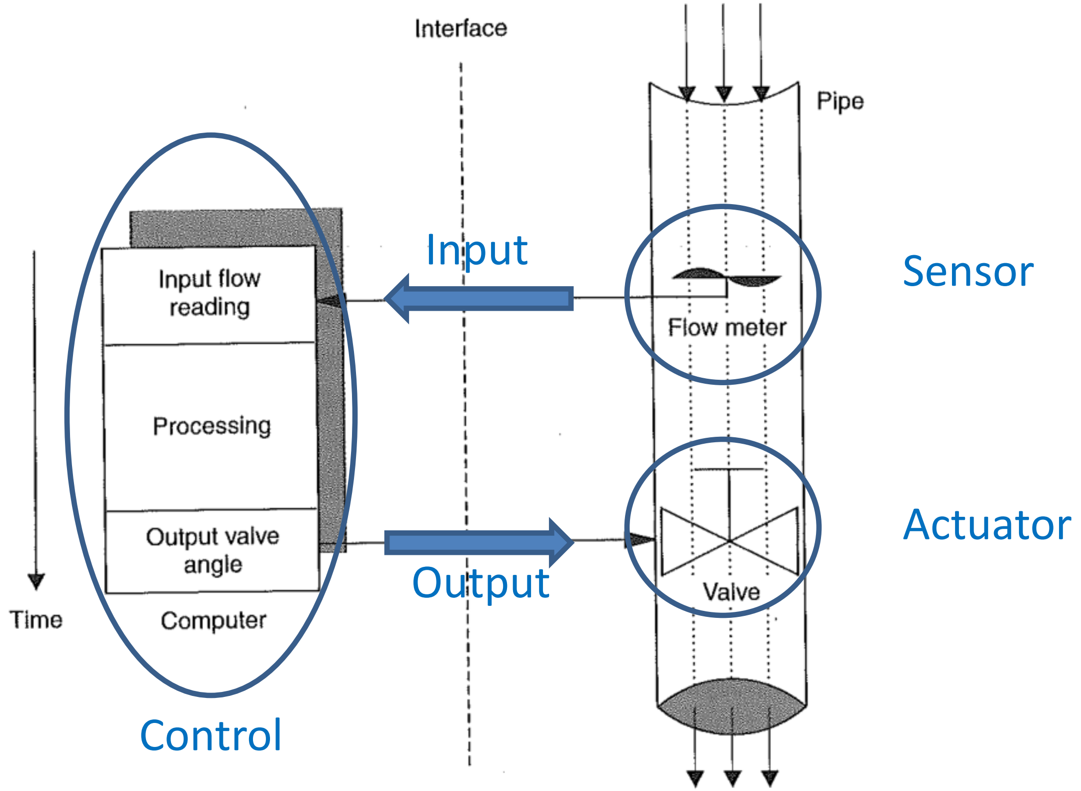


Figure 1.1 A fluid control system.

[Burns09]

Simple real time system(1 component)



Input

Output

Sensor

Actuator

Control

Figure 1.1 A fluid control system.

[Burns09]

Not so simple real time system

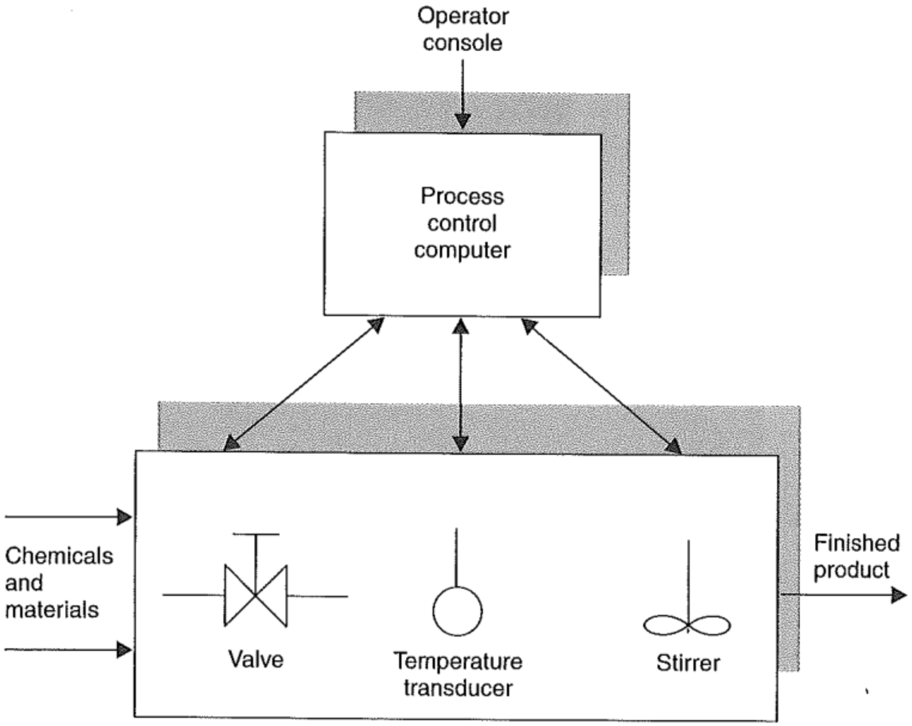


Figure 1.2 A process control system.

[Burns09]

Even more complex

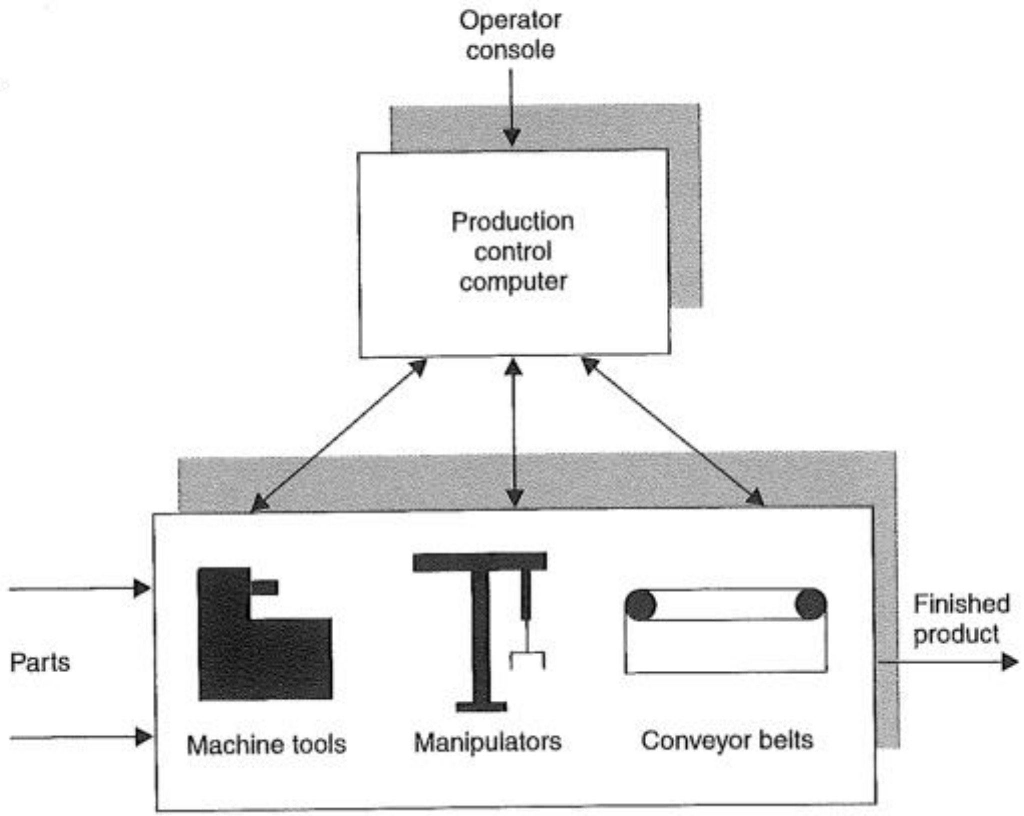
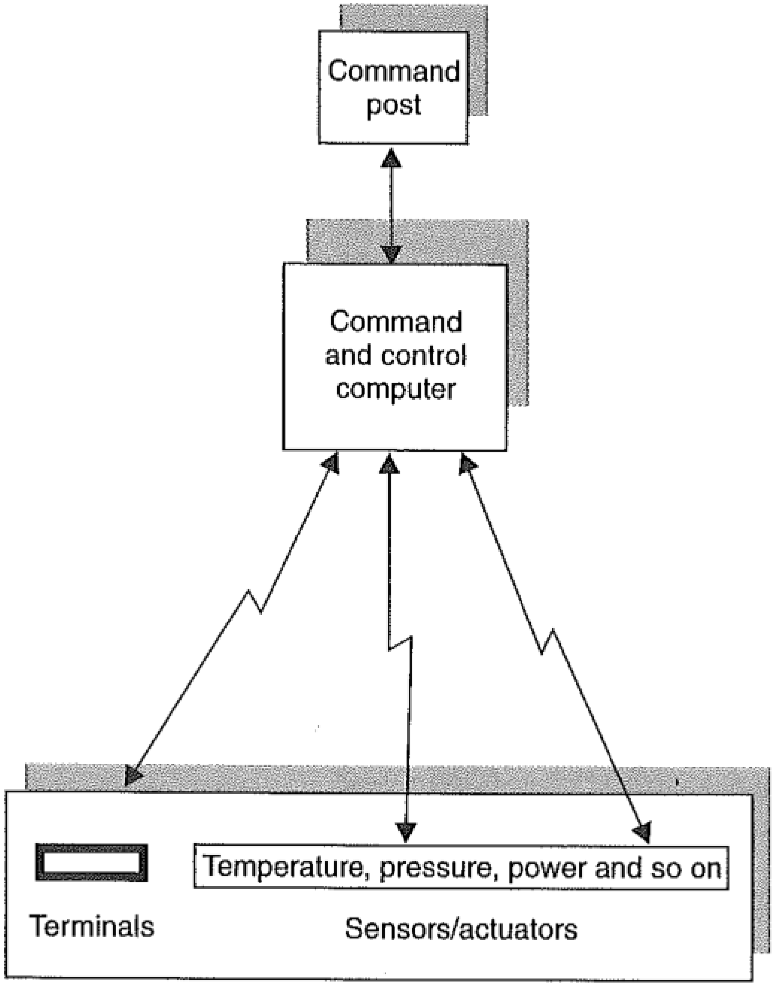


Figure 1.3 A production control system.

[Burns09]

A generic real time system


Command post
Command and control computer
Temperature, pressure, power and so on
Terminals
Sensors/actuators

[Burns09]

Embedded Computer Systems

- Types of sensor/actuators (devices)
  - **time-triggered** → periodic activities
  - **event-triggered** → aperiodic / sporadic activities
- Periodic activities → with a defined cycle time
- Aperiodic → activity started by environment
  - interrupt for example
- Sporadic → aperiodic but with limited occurrances

RTS elements

- Environment
- Real time hardware
- Real time software

RTS: Non functional requirements

- Safety – prevent RTS from coming into harm
- Performance – response time/throughput
- Fault-tolerance – design faults
- Robustness – protect from external actions
- Scalability – perform with added load
- Security – protect from intentional harm

Real time hardware

- Hardware of a RTS must be **predictable**
  - But… instruction pipelining, branch prediction, virtual memory and caching → FAST ☹
- Devices will be connected to the processor through one or more communication busses
  - shared bus → competition
- External communication makes it worse
  - wifi, ethernet, …

Real time hardware

- No requirement to be fast
  - Just **fast enough** to control the expected environment in the desired manner.
- Example: 8-bit ATtiny804/1604
  - 16KiB memory
  - 20MHz. What’s your computer frequency?
  - ~ 50cts (less in bulk)

RTS elements

- Environment
- Real time hardware
- Real time software

RT apps: Challenges

- Cost
- Correctness (close to error free)
- Main memory availability (RAM)
- Code size restrictions
- Processor speed
- Power consumption
- Available peripherals

Index

- Why Real Time Systems?
- What are Real Time Systems?
- RTS types
- RTS architecture
- RTS programming
- Examples

How do we program a RTS?

- 2 options depending on access to resources:

1) **direct** through machine instructions, and
2) **indirect** through an OS that mediates requests

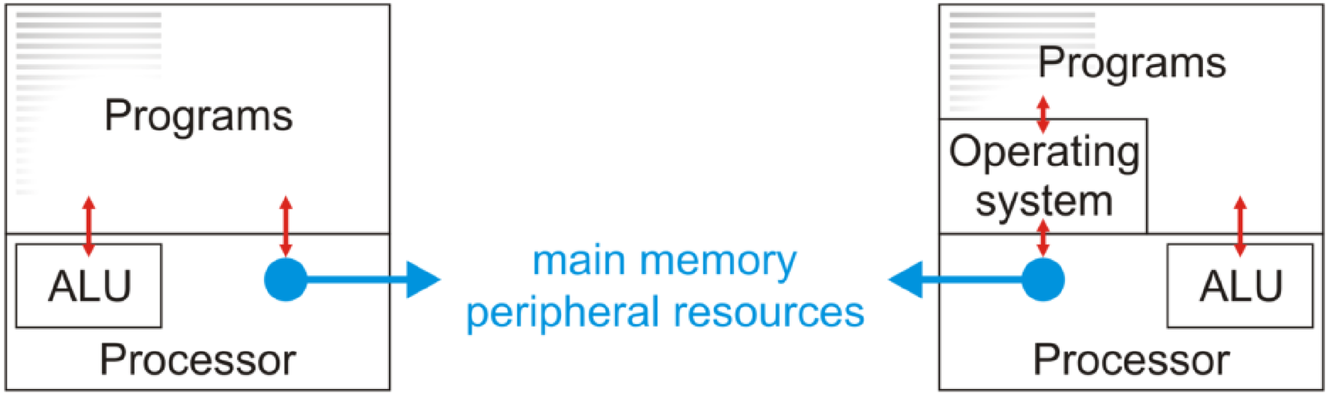


Figure 1-2. Configuration of smaller embedded systems versus larger embedded and general-purpose systems.

[Harder18]

Direct access to resources

- Simplest case may be programmed directly
  - microcontroller
  - low level programming
  - direct access to hardware

- What did you do on embedded systems course?

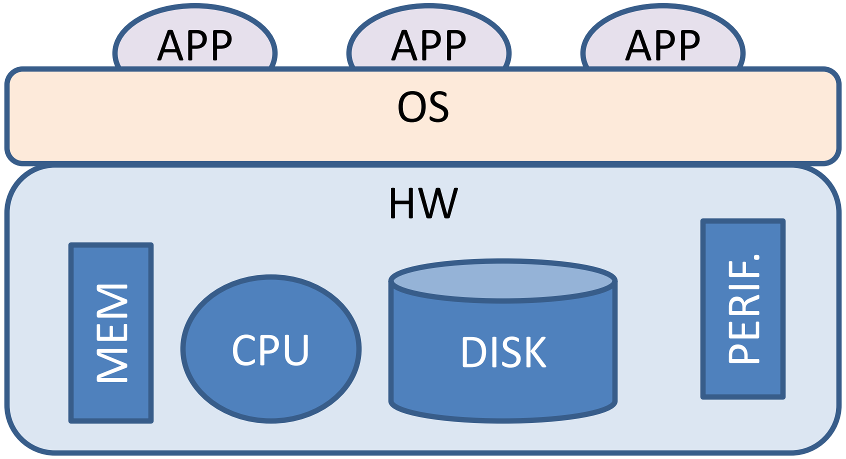
Indirect access to resources

- Complex RTS needs to abstract the system resources → OS with RT characteristics
  - higher level programming

- Do you know what an OS is, right?

Operating System

- “An **operating system (OS)** is system *software* that *manages computer hardware, software resources*, and provides *common services* for computer programs”[wikipedia]



Operating System

- OS is a software
  - manages hardware
    - keeps track of each resource usage
    - decides who gets resource
    - decides how long the resource can be in use
  - manages software
    - keeps track of programs execution
    - decides which program to execute next
  - provides common services
    - for users to manage their programs and access to resources
- Same non functional requirements as before

RTOS

- In most cases, RTOS == OS Kernel
  - Embedded systems → single purpose
    - “general purpose” OS features are unnecessary
  - RTOS gives you control over resources
    - No background processes
    - Bounded number of tasks
  - RTOS gives you control over timing
    - Manipulate task priorities
    - Change scheduling options

Task

- Job of the RTOS is to execute tasks
- A *task* is a process that repeats itself
  - – Loop forever
  - – Essential block of real time software systems
- Code in tasks grouped in *functions*
  - – Readability, Reuse, …

```
while(1) {
        get_data();
        process_data()
}
```

Interfacing the world

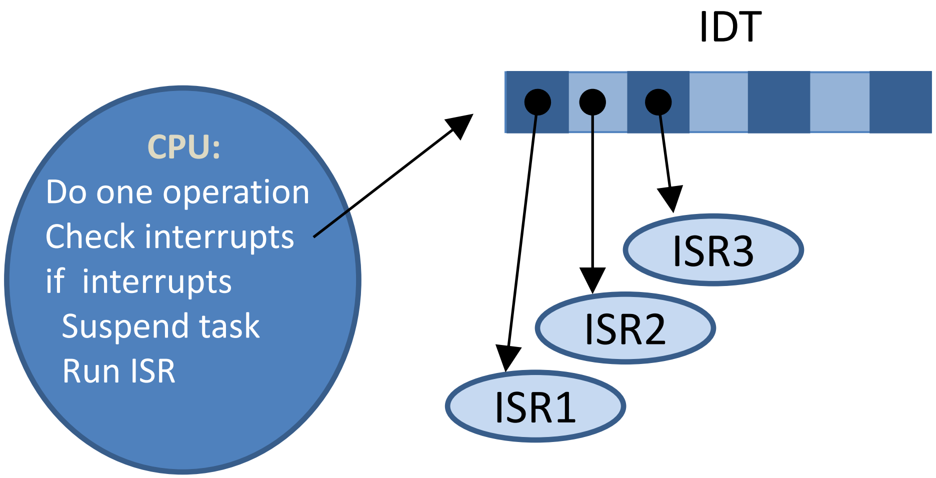
- Methods to access sensors and actuators
  - Polling
  - Interrupt
- The *polling* is the simplest
  - just asking politely for a result/state
    - Lots and lots of times ☺
- *Interrupt* breaks the current flow of execution

Interrupt

- A software/hardware signal
- When CPU receives an interrupt…
  - Completes the instruction being executed
  - Saves the state of the current task
  - Executes the interrupt handler
- Different interrupts may be handled
  - They may be enabled/disabled
  - They may be masked
  - They may be prioritised

Interrupt handling example

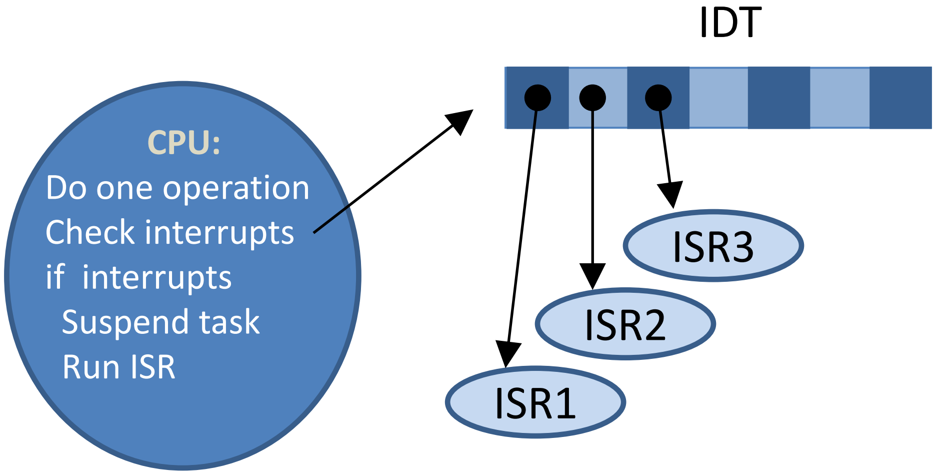
- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)



Interrupt handling example

- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)

task

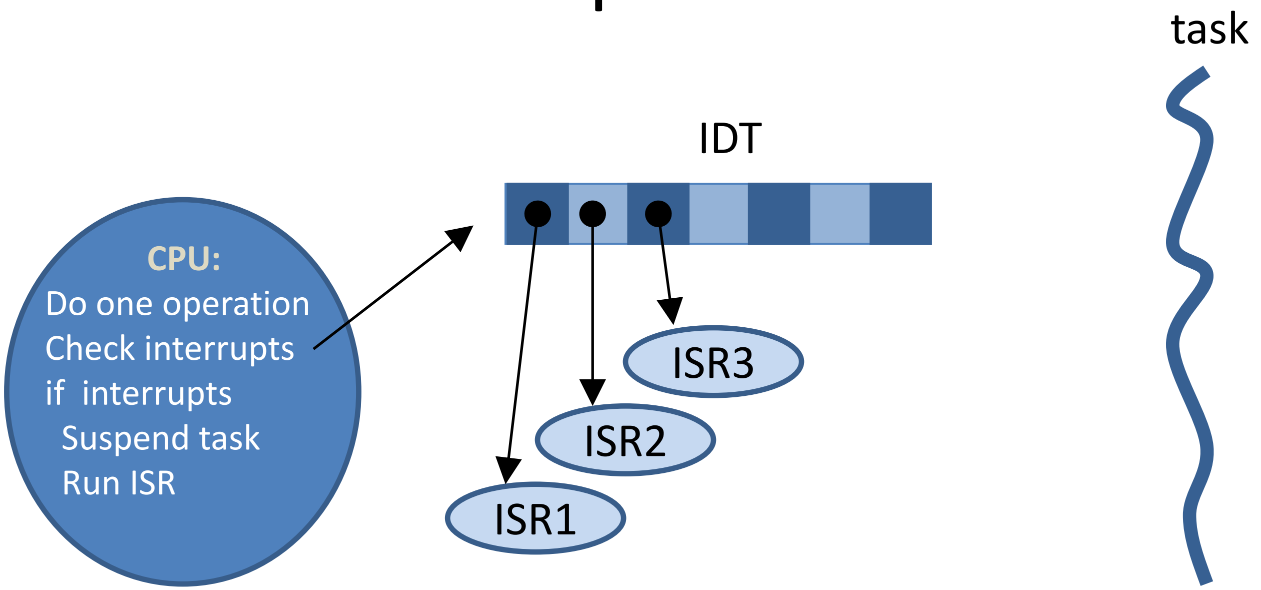


CPU:

Do one operation
Check interrupts
if interrupts
  Suspend task
  Run ISR

Interrupt handling example

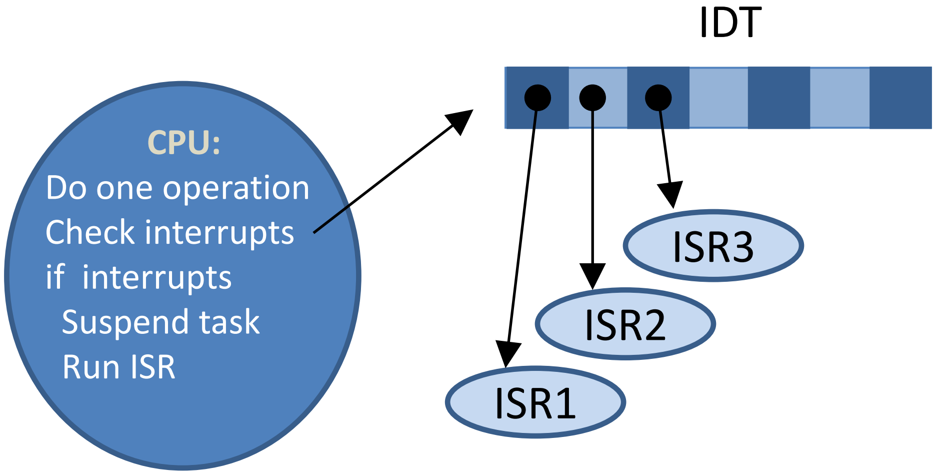
- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)



Interrupt handling example

- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)

task



IDT

CPU:
Do one operation
Check interrupts
if interrupts
  Suspend task
  Run ISR

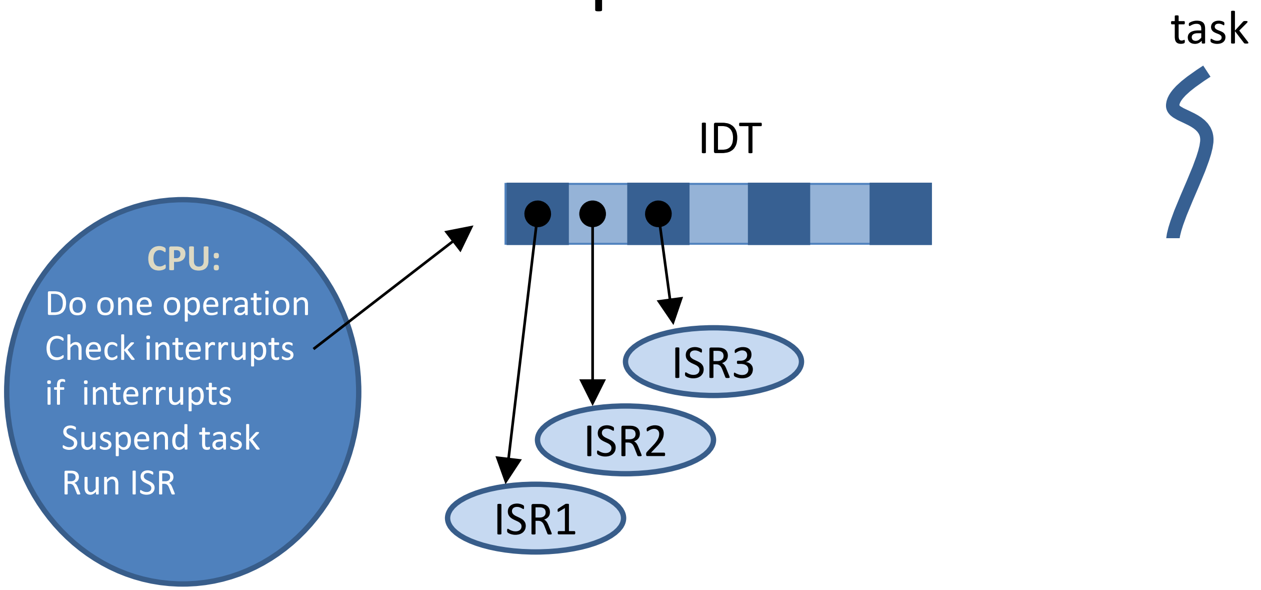
ISR3

ISR2

ISR1

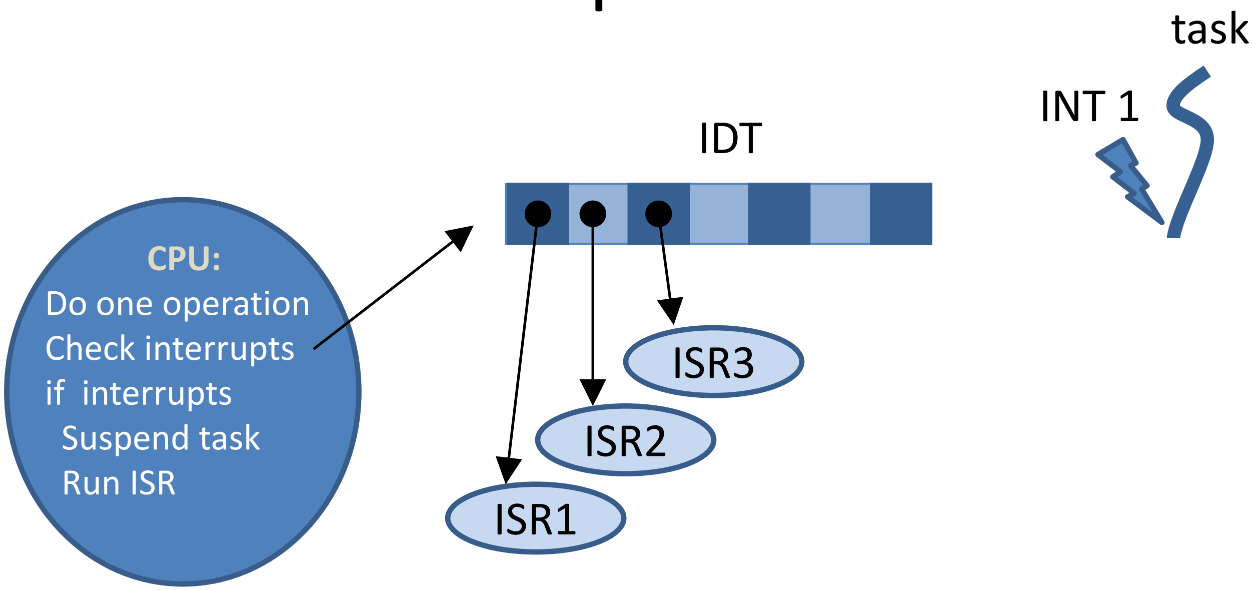
Interrupt handling example

- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)



Interrupt handling example

- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)

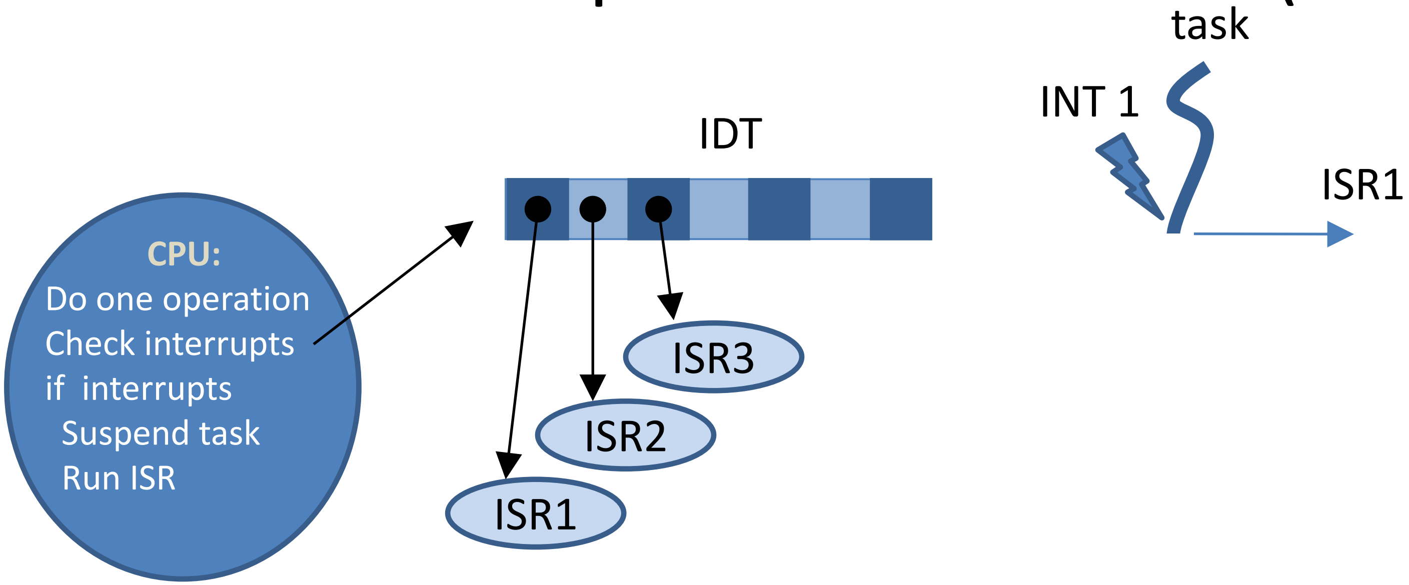


CPU:

Do one operation
Check interrupts
if interrupts
  Suspend task
  Run ISR

Interrupt handling example

- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)



task

INT 1

ISR1

IDT

CPU:

```
Do one operation
Check interrupts
if  interrupts
  Suspend task
  Run ISR
```

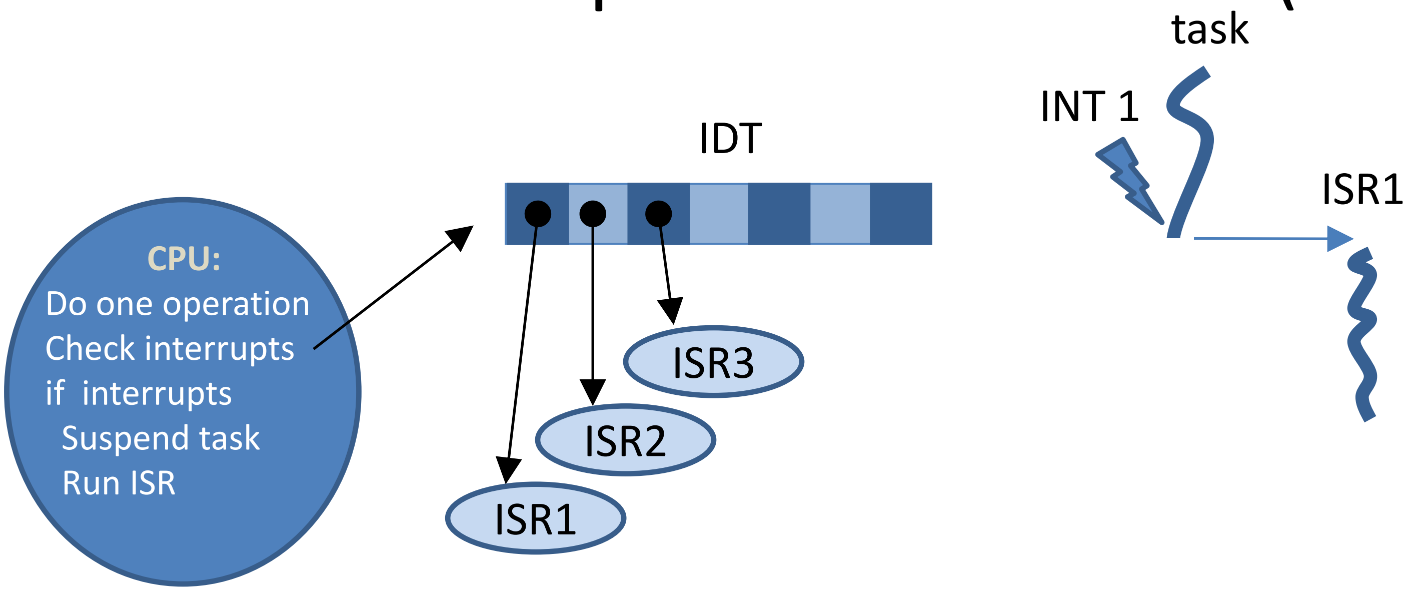
ISR3

ISR2

ISR1

Interrupt handling example

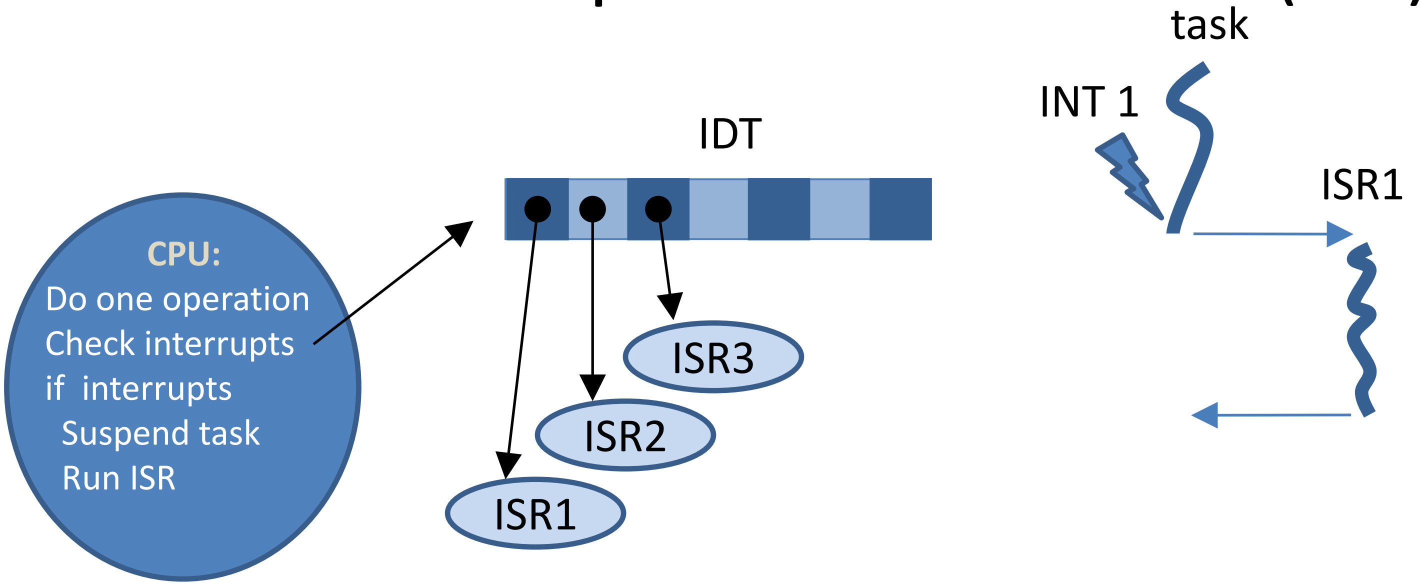
- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)



CPU:
Do one operation
Check interrupts
if interrupts
  Suspend task
  Run ISR

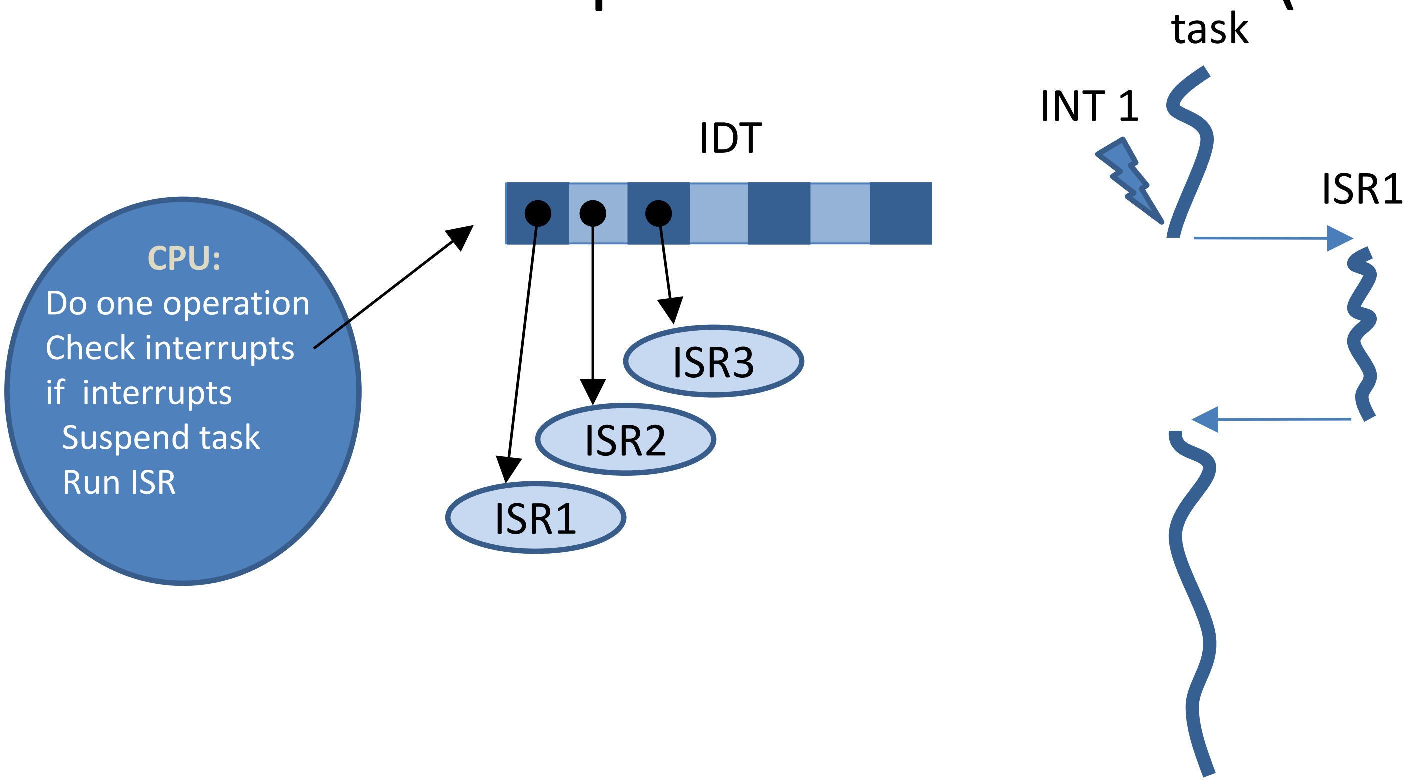
Interrupt handling example

- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)



Interrupt handling example

- Processor checks continually for ints
- Table with Interrupt Service Routines (ISR)



CPU:
Do one operation
Check interrupts
if interrupts
  Suspend task
  Run ISR

Index

- Why Real Time Systems?
- What are Real Time Systems?
- RTS types
- RTS architecture
- RTS programming
- Examples

Why do we need a RTS?

- Volvo commercials
  - – https://www.youtube.com/watch?v=M7FIvfx5J10
    - https://www.youtube.com/watch?v=zn1F-lzoP08
  - – https://www.youtube.com/watch?v=GChq1ywHrw0
  - – https://www.youtube.com/watch?v=Sq0QzErOoag

ESA s Solar Orbiter

- Temperatures > 450°C
- Heat shield
  - MUST be pointed directly at the Sun
- Max deviation +/- 6.5 degrees
- < +/- 2.3 degrees, acceptable for short time
- Takes 40s to reboot

➔50s to react in case of problems!

Aircraft Navigation System

- inputs:
  - x, y, z accelorometer pulses (5ms rate)
  - roll, pitch, yaw angles (40ms rate)
  - temperature (1s rate)
- outputs:
  - compute actual velocity (40ms rate)
  - output velocity to display (1s rate)

→**concurrent** processes with different **rates**

Nuclear Plant Monitoring System

- event triggered by a signal
  - must respond in 1s
- critical signal
  - over-temperature of nuclear core
  - must respond in 1ms

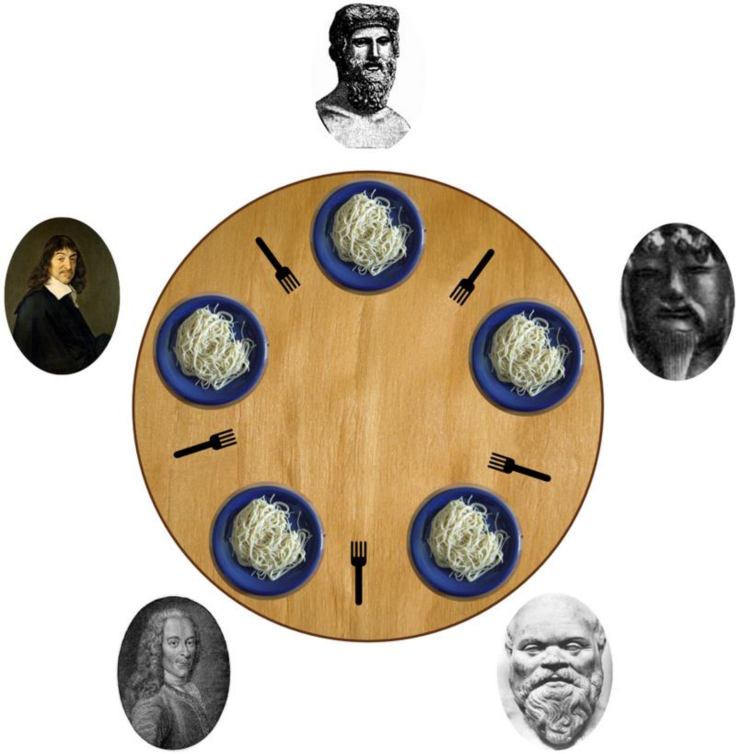
→process with different **priorities**

Dining philosophers

- think
- pick left fork
- pick right fork
- eat
- drop left
- drop right
- think
- …

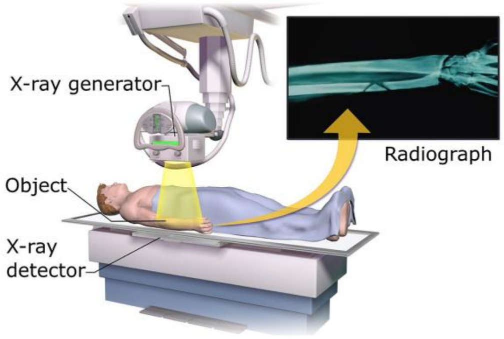
→Deadlock

https://en.wikipedia.org/wiki/Dining_philosophers_problem



Therac-25

- Computer controlled therapeutic radiation machine for treatment of tumors (~1985)
- Massive radiation overdoses
  - 6 deaths and serious injuries
- Caused by race condition



N. G. Leveson and C. S. Turner, "An investigation of the Therac-25 accidents," in Computer, vol. 26, no. 7, pp. 18-41, July 1993, doi: 10.1109/MC.1993.274940.

Therac-25

- 2 modes:
  - electron mode (low energy) and
  - X-ray mode (high energy)
- Operator enters X-ray mode erroneously
  - detects the mistake and quickly switches back, but
    - treatment phase task ignores keyboard input
    - high energy radiation with no indication to the operator

Summary

- RT System concepts
- Categorization of RTS
  - Soft/Firm/Hard
- Architecture of a RTS
  - Control/Sensors/Actuators
- RT Operating Systems
  - Different from GPOS
- Examples showing some challenges
  - concurrency, shared resources, interleaving

References

- [Burns09] “*Real-Time Systems and Programming Languages*”. Alan Burns and Andy Wellings. 2009. Ch.1.
- [Harder18] “A practical introduction to real-time systems for undergraduate engineering”. Harder, Douglas W et al. 2018. Ch.1.
- [Krywko20] “*What operating systems keep things running in space?*”Jacek Krywko. 2020. https://arstechnica.com/features/2020/10/the-space-operating-systems-booting-up-where-no-one-has-gone-before/
- [Butazzo11] “*Hard Real-Time Computing Systems*.” Giorgio C. Buttazzo.2011.