HW and SW technologies for industrial automation

Leonardo Labs
Introduction – Automation system – Control devices

Gianmaria DE TOMMASI
Email: detommas@unina.it

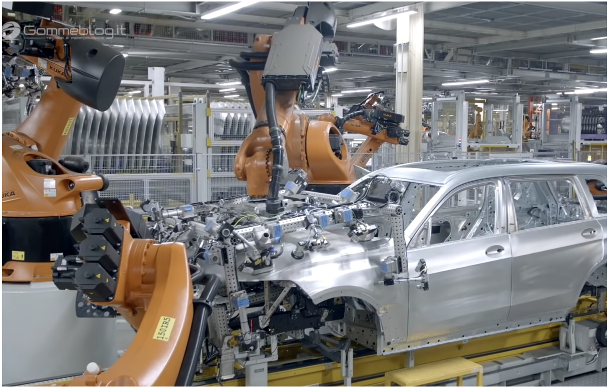
October 2020

DIE TI. UNINA UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE

Outline

What is industrial automation?

- What is industrial automation?
- BMW fully automated car factory (mass production)



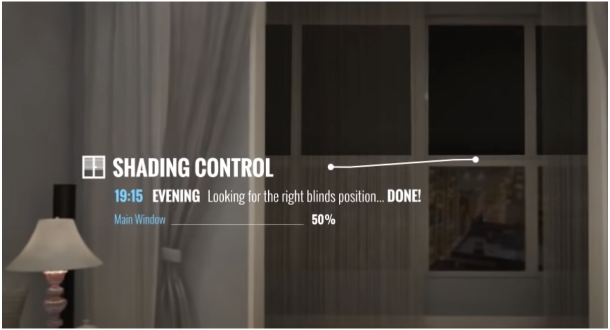
What is industrial automation?

Amazon warehouse (logistic)

What is industrial automation?

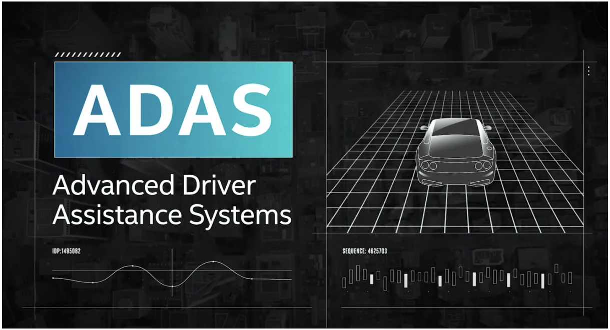
Industry 4.0 (mass production)

Domotics


SHADING CONTROL
19:15 EVENING Looking for the right blinds position... DONE!
Main Window 50%

. . . and more

Autonomous driving



THE ANSWER

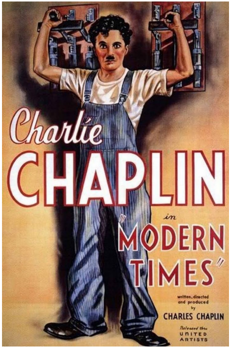
(Industrial) **automation** includes both

- technology
- methodology

by which a process or procedure is performed with **minimal** human assistance

Main benefits of automation

- labor savings
  - reduce the need of tedious and dangerous jobs



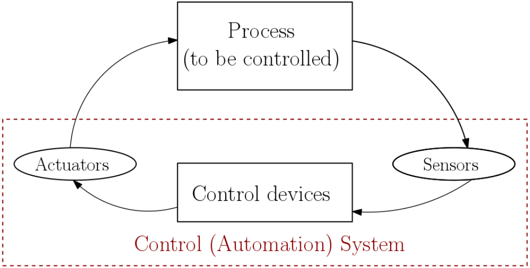
Main benefits of automation

- labor savings
- savings production time (costs saving)
- savings stocks (costs saving)
- savings production waste (costs saving)
- energy saving (costs saving & **key feature for green economy**)
- reduce environmental impact (**key feature for green economy**)

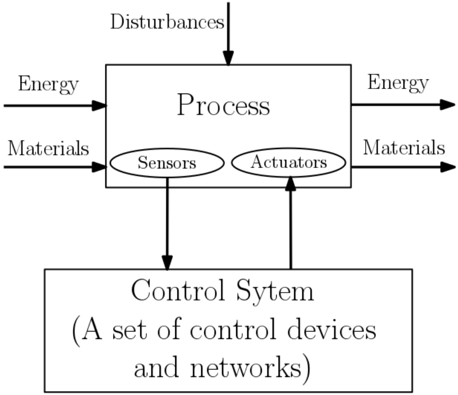
Summarizing: **better use of the resources**

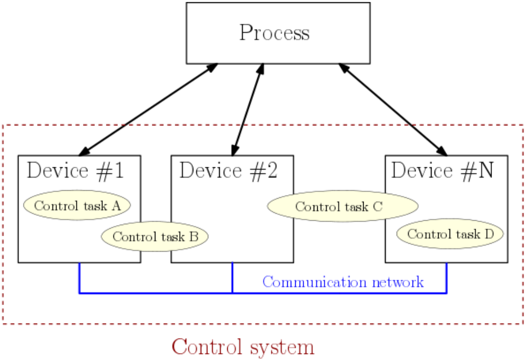
Overview of this 2 days course

- **What we will see during these 2 days?**
  - Control devices for (industrial) automation: main requirements & architectures
  - The IEC 61131-3 standard (Programming languages for Programmable Logic Controller)
    - ladder diagram
    - functional block diagram
    - instruction list
    - structured test
    - **sequential functional chart (SFC)**
- **What tools we will *play* with?**
  - Matlab/Simulink
  - CODESYS `https://www.codesys.com/`


Process
(to be controlled)
Actuators
Control devices
Sensors
Control (Automation) System

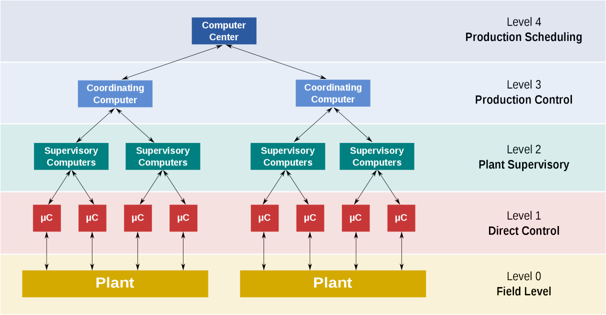
Control systems - *Our* point of view



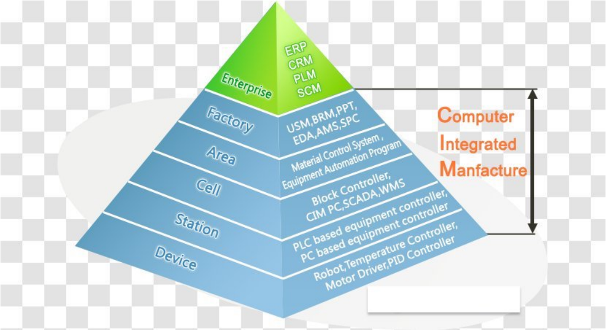


Process
Device #1
Device #2
Device #N
Control task A
Control task B
Control task C
Control task D
Communication network
Control system

Distributed control systems
Hierarchical view



Distributed control systems
The CIM model



- A control device should be able to execute a *user-defined* algorithm. . .
- . . .hence it is a *computer*. . .
- . . . **BUT** it must be able to *interact* with the process
  - via I/O boards and (nowadays) via networks and smart devices

Main functionalities

- *classical* closed-loop control (set-point tracking, trajectory tracking)
- sequential and logic control
- alarms management
- human-machine-interface (HMI)
- communication protocols

Requirements for a control device
Task execution mode

Control tasks can be executed according to three different *execution modes*

- **periodic** (needed for *classical* closed-loop control)
- **cyclic** (main execution mode for sequential and logic control)
- **event-based**

Scalability and robustness

- HW and SW capabilities can be *scaled* according to the need
- Reliable operation in harsh environment (dust, mechanical vibration, electromagnetic pollution , . . .)

- **Control systems are real-time systems**
- A real-time system is a system (hardware+software) subject to *real-time constraints*.
- In a real-time system, the result of a computation is correct if
  - is correct (!). . .
  - . . . AND meets specified time constraints – the so called *deadlines*

Example of non-real-time algorithm

- **Functional requirement:** Given the two weights `w1` and `w2`, compute the weighted sum of the two inputs `u1` and `u2`

```
double weightedSum(double u1, double u2, double w1, double w2){
           double result = w1*u1+w2*u2;
           return result;
}
```

Non-real-time VS real-time - 2/2

Example of Real-time algorithm

- **Functional requirement:** Given the two weights `w1` and `w2`, compute the weighted sum of the two inputs `u1` and `u2`
- **Non-functional requirement:** perform the computation in **at most** 1 ms

Now writing...

```
double weightedSum(double u1, double u2, double w1, double w2){
            double result = w1*u1+w2*u2;
            return result;

}
```

...is no more sufficient to fulfill the requirements!

We should exploit (indirectly) the hardware architecture and (directly) the operating system, in order to meet the time constraint

Time constraints 1/2

- A computation must be performed **every X time units**
  - is a *periodic* activity (task), and the time constraint must be met with a given accuracy (*jitter*)

Examples

- “the control action to be applied by the aerosurfaces of an aircraft must be computed every 5 ms"
- “System *A* must send a message to system *B* every 10 s"
  - **Remember: real-time does not necessarily means *fast*!**

- A computation must be completed **within Y time units after its triggering**
  - is a task with a *deadline* (*cyclic* or *event-based*)

Examples

- “the cyclic execution of a PLC must terminates within 200 ms"
- “stop the cruise control within 50 ms after the break press"

- **Note: usually a periodic task should also meet a deadline**

Hard and soft real-time

- **Hard** real-time systems
  - **Missing (even a single) deadlines means system failure (!)**
- **Safety critical** systems
  - Missing deadlines can cause serious loss
- **Soft** real-time systems
  - Deadlines may be missed and mainly cause a deterioration of the QoS

- Real world (real-time) systems have a mix of hard/soft components
- The distinction between hard and soft real-time is somewhat subjective
- **Soft real-time is not Non-real-time (!)**

Assess schedulability

- Given *n* real-time tasks...
- ...given the correspondent time constraints (deadlines)...
- ...given the hardware (and software) architecture...
  - is it possible to meet all the timing requirements, i.e. is it possible to schedule the tasks?
    - Are the deadlines met for all the cyclic and event-driven tasks?

      `End_time(task_k)-Start_time(task_k)` $\leq$ `Deadline(task_n)`
    - Are the periodic tasks executed with the required accuracy? Do they meet their deadlines?
- There exist formal methods that permits to assess schedulability (under given assumptions)

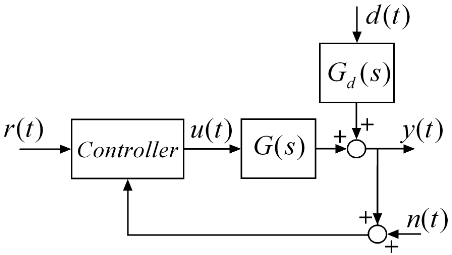
Real-time operating systems (RTOS)

- Interrupts/Polling
- Multitasking (concurrency)
- Timer support
- Static scheduling/Preemptive scheduling (priorities)
- Task Segregation
- . . .

Some RTOS

- WindRiver VxWorks
- QNX Neutrino
- RTAI (Linux patch)
- FreeRTOS
- Windows CE

Example - Continuous control system



The plant

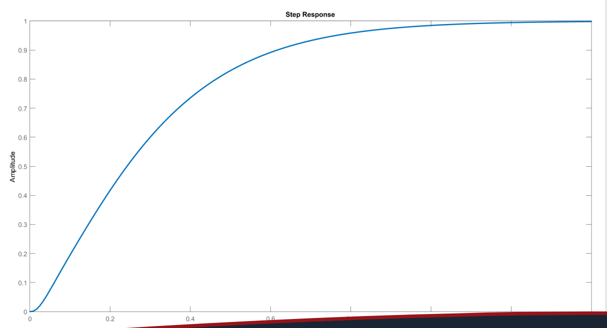
$$G(s) = \frac{2.5 \cdot 10^5}{(s+10)(s^2+80s+2500)}$$

Leonardo Labs Aerotech Campus ACADEMY

The continuous-time controller

$$C(s) = \frac{2.24(s+25)^2}{s(s+200)} \tag{1}$$

Open-loop step response



The discrete-time controller

Given the sampling frequency $f_s = 200$ Hz, the Tustin approximation of the controller (1) is

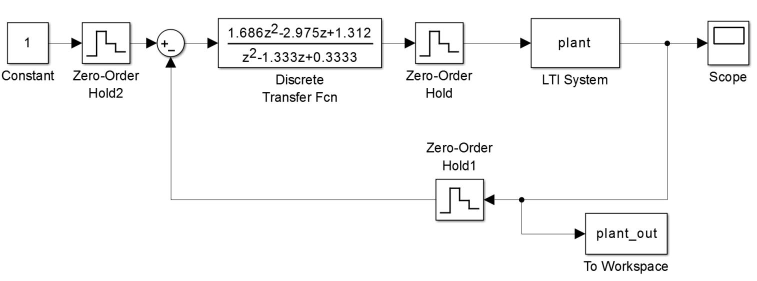
$$\hat{C}_d(z) = \frac{1.686(z-0.882)^2}{(z-1)(z-1/3)} \tag{2}$$

- Implementing the discrete-control law (2) means
  - **Functional requirement:** to write a task that computes the correspondent difference equation
  - **Non-functional requirement:** to execute the task every 5 ms (assuming negligible execution time)

To meet or not to meet (the deadlines)?

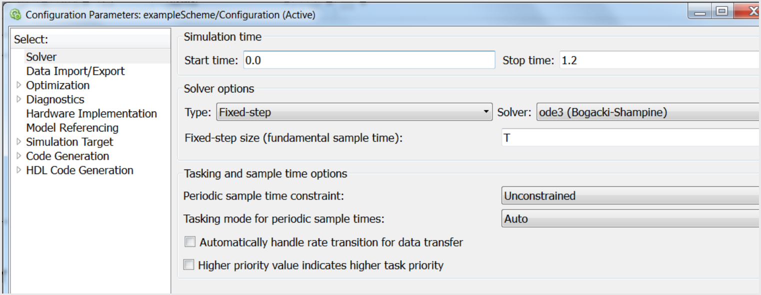
Leonardo Labs Aerotech Campus / ACADEMY

Use Simulink. . .



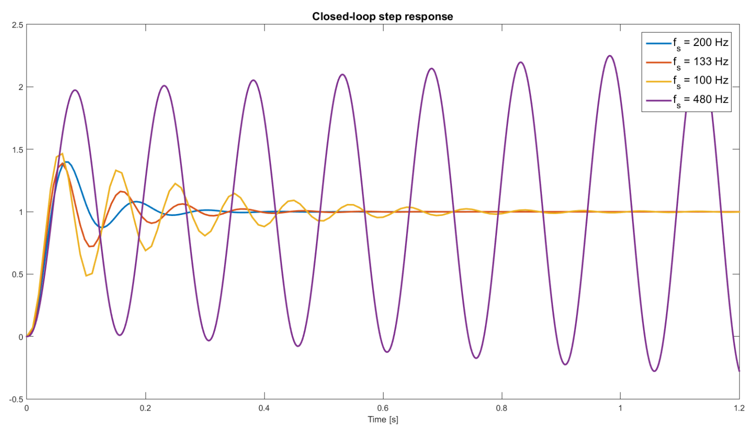
To meet or not to meet (the deadlines)?

. . .with **Fixed time-step** solver. . .



Leonardo Labs Aerotech Campus / ACADEMY

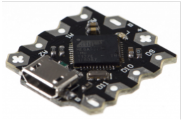
. . .changing the time step

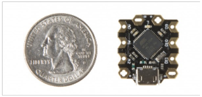


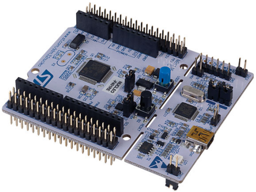
Control devices architecture

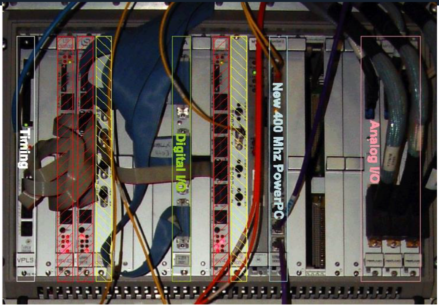
- monolithic
- bus-based
- PC-based
- cloud-based

Monolithic devices - Examples









- VME architecture
- PowerPC 400 MHz
- 512 MB RAM
- ATM (for real-time comms) and Ethernet (for non-real-time comms) network interfaces
- VxWorks OS
- Sampling frequency 500 Hz

Bus-based architectures - Examples

Leonardo Labs Aerotech Campus / ACADEMY

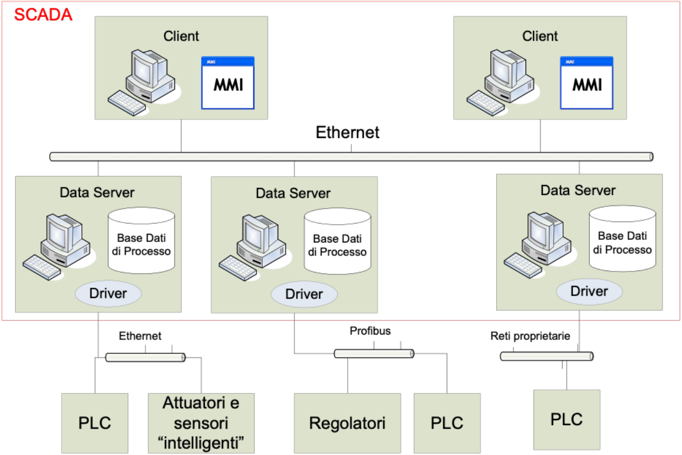
- Bus architecture based on ATCA+PCIe
- Multi-core processor (Inter Core2 Quad)
- Linux+RTAI OS
- 192 signals acquired by ADCs (18 bits 2 MHz) and transferred at each cycle
- 50 μs control loop cycle time with jitter $< 1 \ \mu s$
- Always in real-time (24 hours per day)
  - 1.728×10^9 50 μs cycles/day

Leonardo Labs Aerotech Campus / ACADEMY

The Programmable Logic Controller (PLC)

The IEC 61131-3 standard defines the software architecture and the programming language of a control program in a PLC system.

Supervisory Control and Data Acquisition Systems (SCADA)



HW and SW technologies for industrial automation

Leonardo Labs

Introduction – Automation system – Control devices

Gianmaria De Tommasi
Email: detommas@unina.it

October 2020

UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELLE TECNOLOGIE DELL'INFORMAZIONE