ELEC-H410
REAL-TIME SYSTEMS

Chapter 1

Introduction

ELEC-H-410

CONTENTS

ELEC-H410 : Introduction

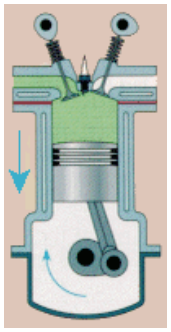
CONTENTS

- **example: ignition of an internal combustion engine**
  - **electro-mechanical**
  - **introdution of electronics**
  - **current computerized ignitions**
  - **decentralization and network**
- concept of real-time
- specificities of embedded systems

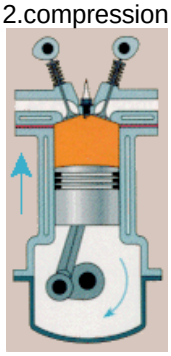
Spark-plug ignition of ICE

4-stroke cycle: a spark must occur at the right moment

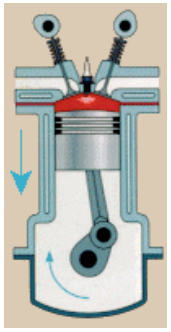
1.intake



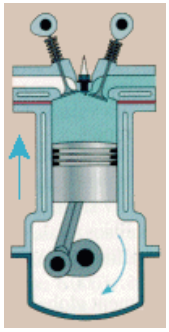
2.compression

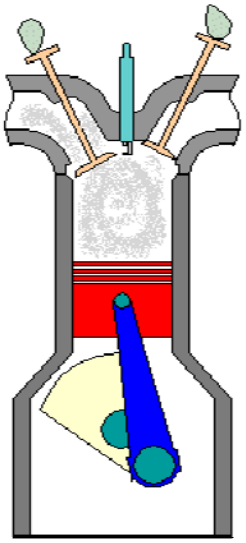


3.combustion



4.exhaust

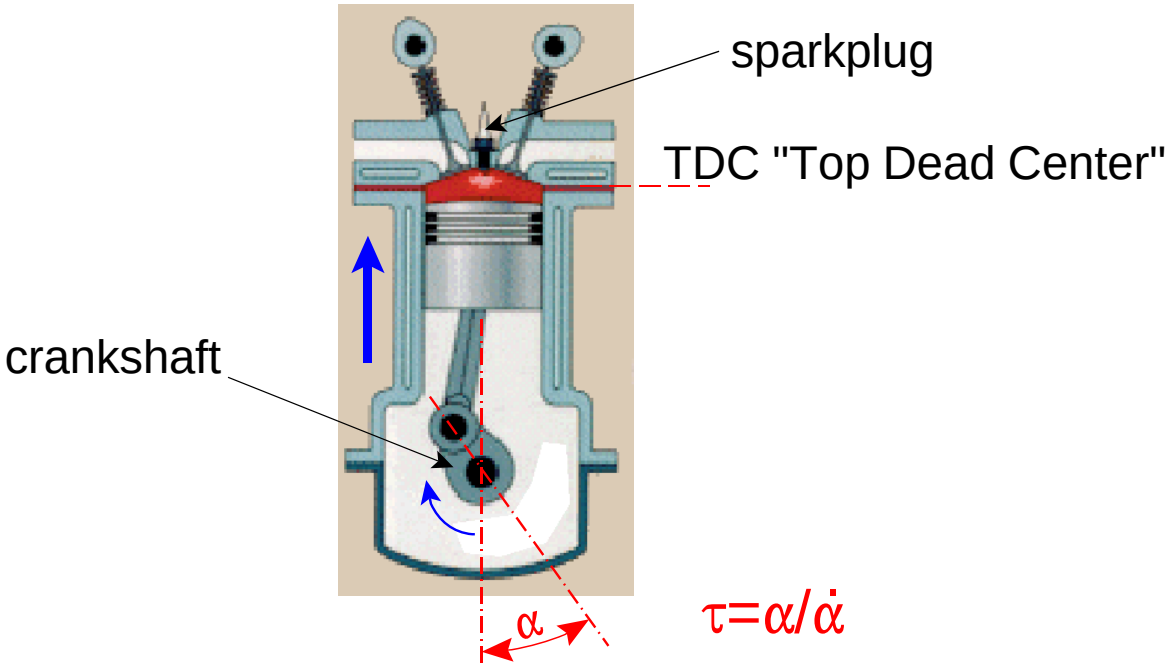




A good example to illustrate the embedded systems and their evolution during last years 40 years is the spark ignition of the internal combustion engines. This figure is a simple recall of the well-known four-stroke cycle.

Ignition timing

∃ time and angle delay before TDC



Let us focus here on the **ignition timing** or **ignition delay**.

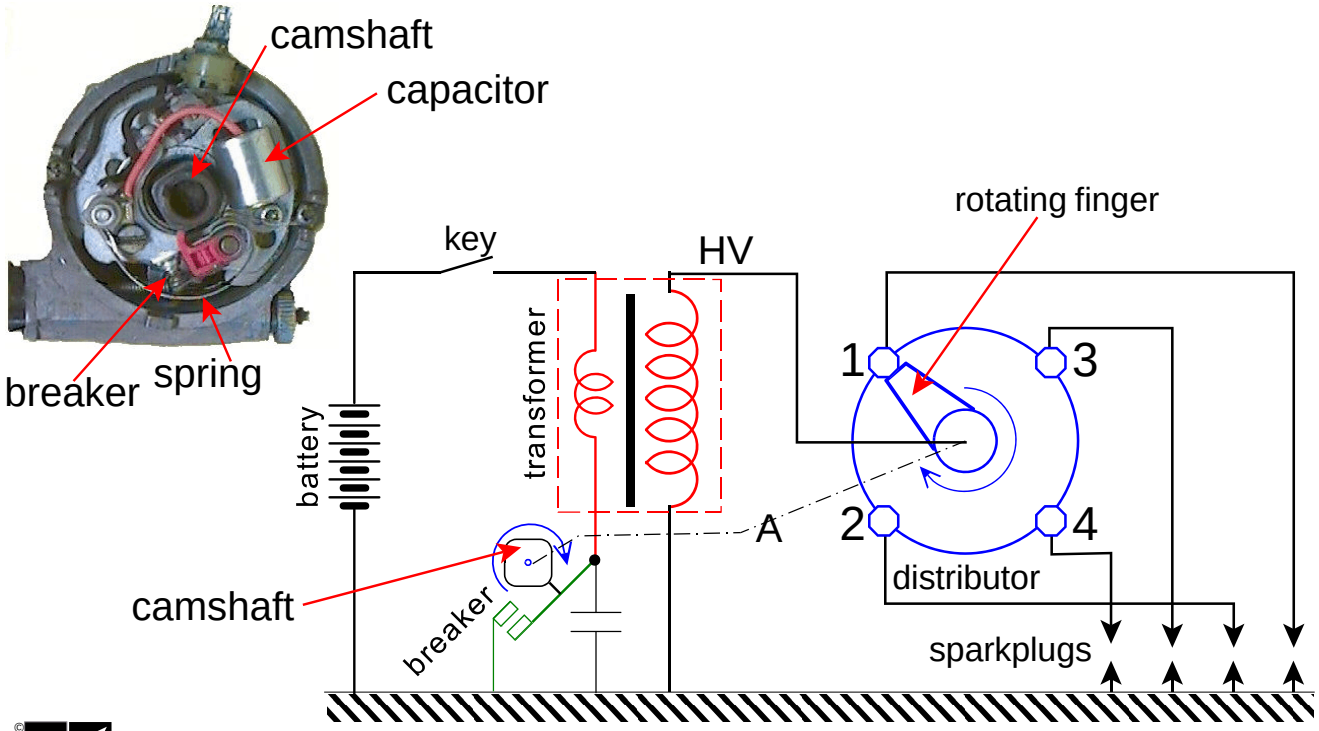
The spark must be delivered by the sparkplug before the piston reaches to the top of its compression stroke (called TDC *Top Dead Center*), because the propagation of the flame is not instantaneous. The purpose is to get the maximum force on the piston just when it begins its descending stroke.

The ignition delay is expressed as an **"advance angle" α** corresponding to the rotation of the crankshaft between the ignition and the TDC.

In the time domain, It corresponds to an **ignition delay τ** proportional to α and inversely proportional to the actual angular speed.

Electromechanical solution (1900-1970)

synchronous mechanical breaker + distributor



Until the 70ies, the ignition was based upon an electromechanical solution.

During compression, a mechanical switch (the breaker) is closed and applies the voltage of the battery to the primary winding of the transformer.
The secondary of the transformer is open so that the primary (=magnetizing) current rises linearly.

Th drawing illustrates the moment when the crankshaft arrives at the ignition angle
- a camshaft rotating synchronously (at half of the motor speed) opens the breaker
- to ensure the continuity of the magnetic flux, a current should raise in the secondary winding
- this is possible because the HV lead coming from the secondary is connected to a rotating finger moved by the same camshaft. At this moment the finger in just in front of a contact (1) connected to the ignition lead of the sparkplug 1. Thanks to the high number of turns of the secondary, the voltage is sufficient to cause two arcs
  - the first one between the finger and the contact (1)
  - the second one between the two electrodes of the sparkplug , causing the ignition

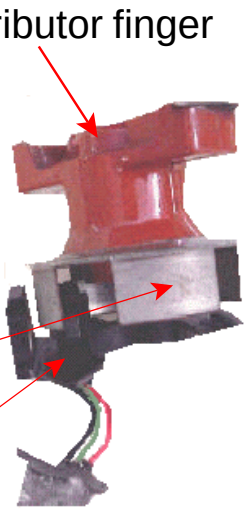
REM
- the camshaft is in fact an axis A presenting one embossing per cylinder (4 in this figure) and turning at half the speed of the crankshaft. The finger of the distributor is placed on the same axis A which ensures the mechanical synchronisation between the opening of the breaker by the camshaft and the passage of the finger in front of the contact of the right plug
- a capacitor in placed in parallel with the breaker to partly absorb the arc occurring before the secondary current appears

This is in fact an **electromechanical real-time system** .

Transistorized ignition (1970+)

a first step to modern ignition

- ☹ **wear of the breaker contacts**
  - capacitor helps but not sufficient
  - frequent trimming (<5000 km)
  - limited lifetime (20000km)
- solution: breaker replaced by transistor
  - no arc - no wear
  - electrical signal required to synchronize the driver of the transistor
  - => TDC ***sensors***
    - variable reluctance by rotation of
      - star-shaped magnets
      - ferromagnetic screens
    - Hall sensors or magneto-resistances



Despite the presence of the capacitor, the contacts of the breaker wear, because a small arc remains, always in the same direction, and transports metal from one contact to the other. A crater grows on a contact and a bump is formed on the other one. After some time, it is not acceptable anymore, because the distance between the contacts becomes badly defined, which alters the value of the ignition angle. It is thus necessary to periodically (for example every 2500 km) file the contacts, and to trim the gap which separates them. The lifespan of the contacts was 10.000 to 20.000 km.

At the beginning of the 70ies appeared the first transistorized ignitions, in which the mechanical breaker is replaced by a BJT transistor. This transistor has to be driven by an electrical signal synchronized to the rotation of the crankshaft. Therefore, we need an angular position sensor. Most of such sensors are based upon a magnetic circuit in which an important variation of reluctance occurs to trigger the ignition. Several systems are used:

- a star-shaped magnet
- a magnetic circuit with a permanent magnet and a gap; a "bell" in ferromagnetic material is fixed to the distributor finger. Holes in the bell cause the variation of the reluctance

The variation of the reluctance changes the induction field B, which is detected by a Hall sensor, or a magnetoresistance.
Magnetic sensors are quite robust and insensitive to dust, oil, or grease.

Variation of the ignition timing

depends on load and on rotation speed

- 3 cases
  - ideal: combustion gives a max push just after TDC
  - too early: "knocking": piston receives a shock while still ascending; high risk of destruction
  - too late: loss of power, temperature rises
- variable ignition $\measuredangle$ required because
  - combustion takes more time at low load
  - for a given delay $\measuredangle$ ↗ if rotation speed ↗
- how to change $\measuredangle$
  - by rotating the platform carrying the breaker (or the sensor) relatively to the axis of the distributor

Actually, the ignition advance is a **time delay** which must be adjusted according to the duration of combustion.

- Ideally the advance must occur "just-in-time" so that the maximal pressure of the combustion gases occurs when the piston has just begun its descending stroke, i.e. after the TDC
- If the ignition comes too early "knocking" occurs (which can be heard or sensed by accelerometers) because the piston receives a shock while it is still ascending. This can destroy the piston after a while.
- If the ignition comes too late, part of the force on the piston is lost. The efficiency of the motor drops, while its temperature rises.

A double problem occurs

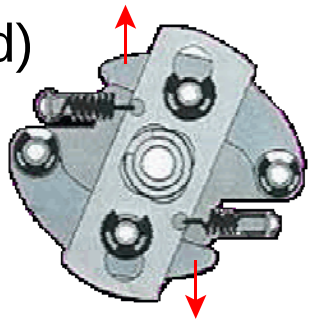
- the propagation of the flame and the combustion is slower at low loads (less fuel in the chamber) and hence the **time advance must be increased at low loads**
- the mechanical synchronization is based upon an angle and not a time, hence, for a given ignition delay, the **ignition angle must be proportional to the angular speed**

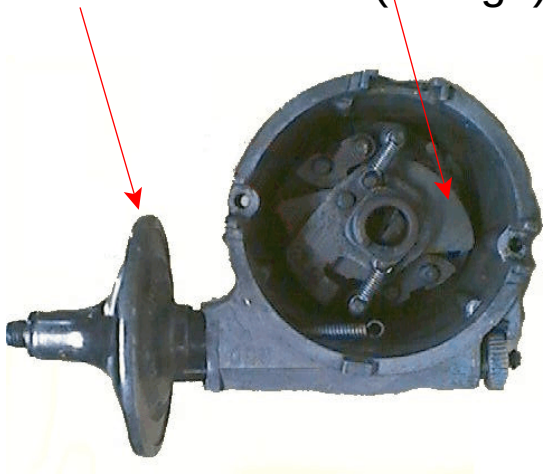
This is achieved thanks to a relative angular displacement of the breaker (or the sensor) compared to the camshaft. This displacement will depend on the load and on the rotation speed.

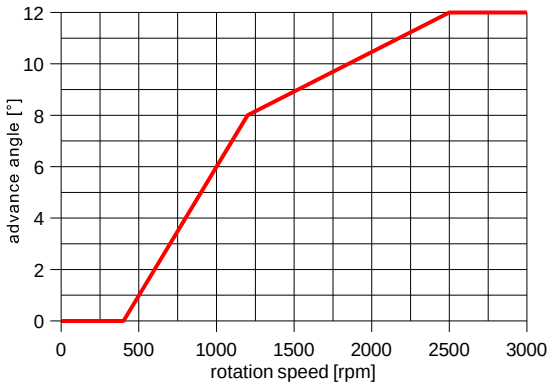
Variable ignition angle

mechanical solution: centrifugal(ω) + vacuum(load)

- 1900: manually (lever on the dashboard)
- automatic control
  - centrifugal advance f(rotation speed)
  - vacuum advance f(charge)





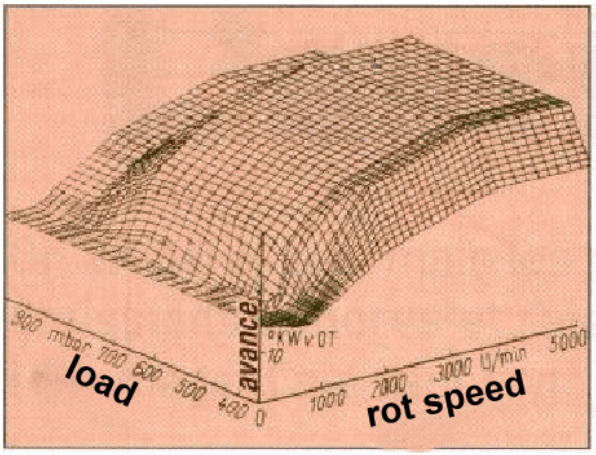


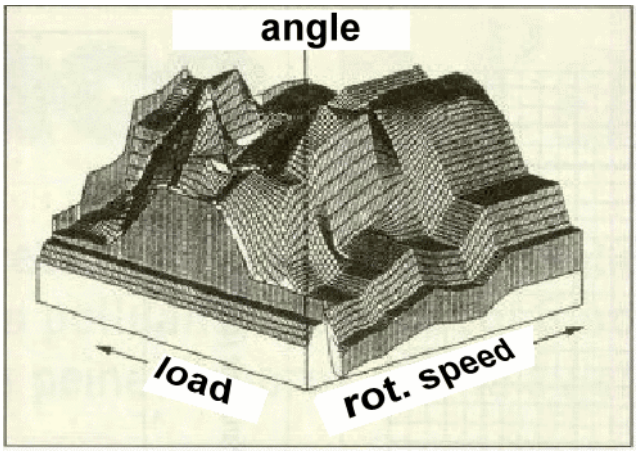
At the beginnings of the 20th century, the adjustment of the advance was carried out manually by a lever in the dashboard.
This was replaced by a purely mechanical automatic control.

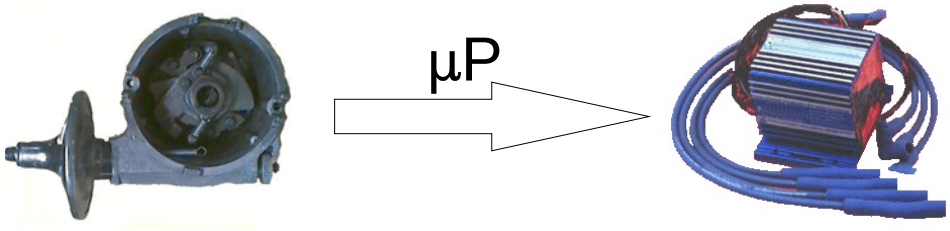
- **vacuum** advance: the load in measured by the depression in the intake pipe. When you press on the accelerator pedal you open the throttling valve and reduce the depression and the advance
- **centrifugal** advance: 2 masses rotating with the camshaft are pulled apart by the centrifugal force; this movement is transmitted by levers to displace the breaker; the advance angle raise with the rotation speed with a 2 or 3 segment piecewize linear law.

Computer controlled ignition

more and more complex timing advance maps





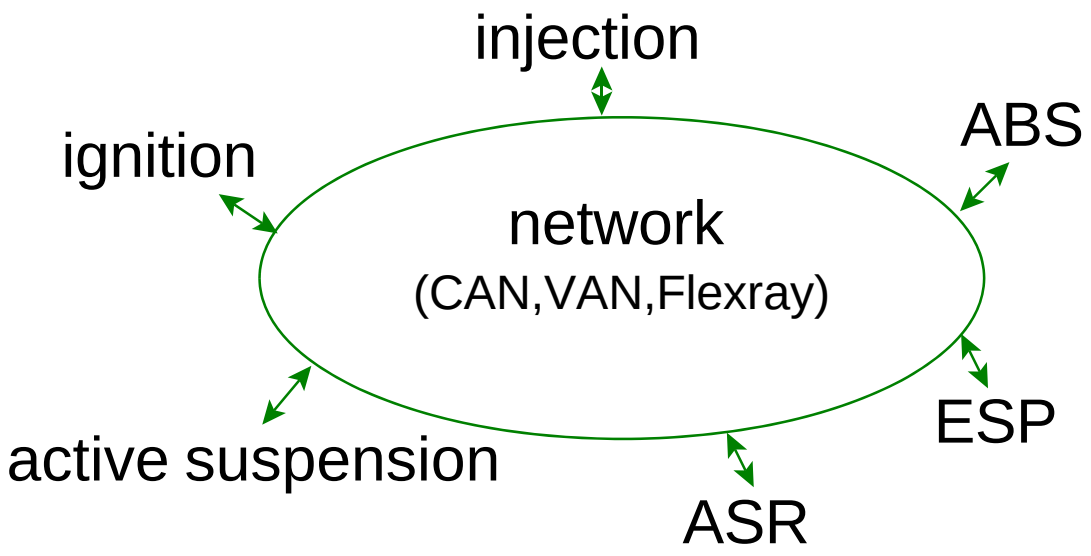


By combining the centrifugal advance with the vacuum advance, we obtain a fairly complex surface illustrated on the left figure.

This is not sufficient to meet the present specifications relative to efficiency and reduction of the pollution. More thorough optimization asks for much more sophisticated laws (right figure) which can only be achieved by tables and algorithms implemented by **powerful microprocessor systems**.

Ignition is only part of the system

interaction via a real-time network



A car is also a perfect process to illustrate **multitasking** and **decentralized** control. Several digital controllers are present:

- ignition
- fuel injection
- ABS (anti-lock braking system): prevents wheel locking during braking
- ASR/TSC (Anti-Slip Regulation / Traction Control System): prevents loss of traction
- ESP/ESC (Electronic Stability Program/Control): detecting and minimizing skids
- Active suspension
- .....

All those controllers have to interact, for example
- there are strong relations between ignition and fuel injection
- when skidding is detected, it can be solved by a combination of action on the brakes and on the accelerator pedal

The interaction is carried out through a special **real-time network**; one of the most famous is **CAN** (Control Area Network) to which a special chapter will be devoted and which you will use during the labs.

Ignition

summary : evolution and involved disciplines

- automatic control (mechanical)
  - the driver does'nt need to be a mechanic any longer
- automatic control + analog electronics
  - performance ↗ , precision ↗, maintenance-free
  - requires sensors
- automatic control + analog electronics + µP
  - complex control laws
  - stability (with time and temperature)
  - decentralization (multi-processor)
  - networking of the various controllers

real-time embedded system

In short, though mechanical automatisms have done a good job during almost one century, the contribution of analog electronics, digital electronics and modern automatic control is crucial to improve efficiency and reduce pollution.

Automotive

an example of real-time system

- periodic well-known tasks
  - ignition
  - injection
- aperiodic (random) events
  - braking for a crossing pedestrian
  - whell falling in a pothole
  - skidding on mud, ice or snow

These systems and their communications must function in real time, i.e. at the speed of the physical,phenomena which they control (speed, position).
Tasks are a mix of
- know events generally associated to periodic tasks like injection or ignition
- perturbations or events that occur asynchronously (randomly): knocking, pedestrian, potholes, ice

ELEC-H410 : Introduction

CONTENTS

- example: ignition of an internal combustion engine
- **concept of real-time**
- specificities of embedded systems

Concept of "real-time"

some definitions

- "system that must react to the external events within the ***time limit*** in 100% of the cases"
- "process ***cannot wait***"
- "system whose ***response time*** is as significant as the quality of operation"
- "capacity to answer a request in a ***suitable time*** to produce a ***suitable reaction***."
- "a right result, but coming ***too late***, is a ***false result***"
- "device, composed of hardware and software, which can cause ***catastrophic consequences*** on the controlled process ***if they exceed their expiries***"
- "aptitude of an application to answer requests of the controlled environment, in ***reaction times compatible with the dominant time-constant*** of this environment"
- "capacity of an application to being able to apprehend a flow of ***asynchronous events*** resulting from a process, without missing any one of these events and to treat each one of them in a determined amount of time"

ordinary system: fulfill tasks + wishes for a mean response time
real-time system: fulfill tasks AND all time constraints

The concept of real-time is of course the "red wire" this course. Some definitions are presented on this slide.

To summarize them: the real-time system must be not only functional, but must also meet specific timing constraints, often essential, sometimes critical.

Concept of real-time

real time != short response time (1)

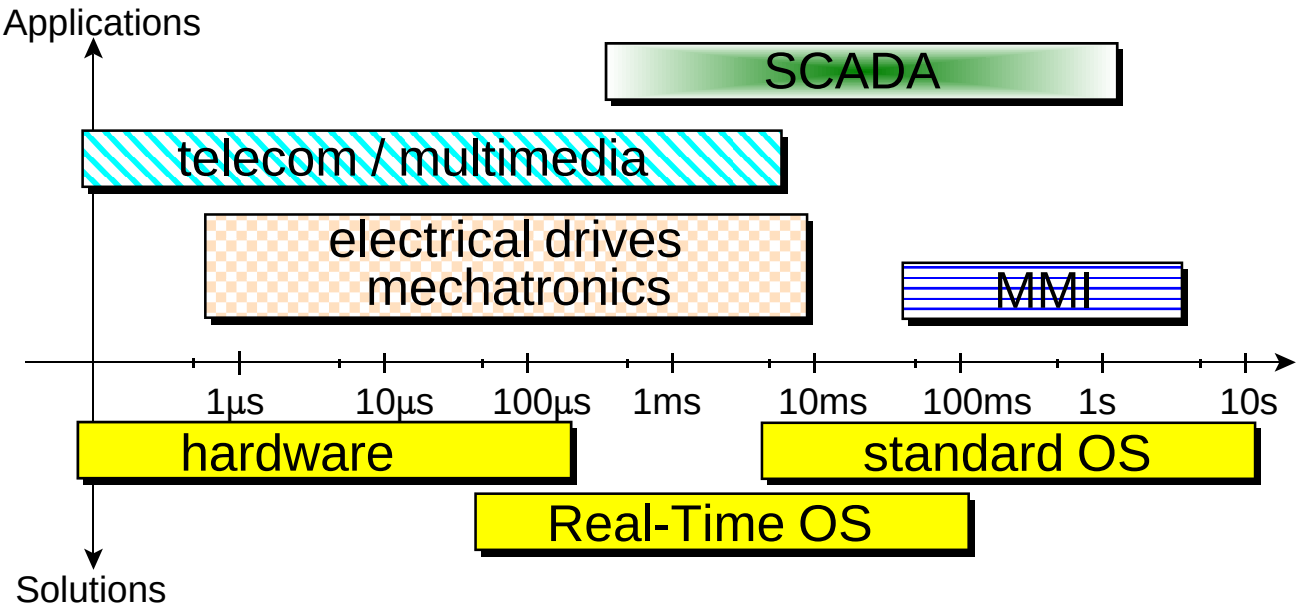
- program to pay salaries
  - bank transfer between 8h and 20h on the last day of each month
  - 12h response time is acceptable.
- avoiding an obstacle in a mobile robot
  - 500ms response time
- nuclear plant
  - overheating detection
  - 2ms to take into account, start cooling and drop the neutron absorbers
- short-circuit in a power transistor converter
  - 10µs to switch-off the gate before the components die

real-time => determinism and reproducibility

The concept of real-time should not be confused with the concept of response time. This slide shows us four examples of real-time system, with response times decreasing from 12 hours to 10 microseconds.
What makes them all real-time systems, it is the existence of **timing constraint (deadlines)** and the need for meeting them in a **deterministic** and **reproducible** way.
You can also notice that the consequences of a missing a deadline go from the **dissatisfaction** to the **catastrophe**.

Concept of real-time

real time != quick response time (2)



This diagram shows some typical applications and the solutions which are applicable for them, according to the scale of reaction time.
Some details on these applications:

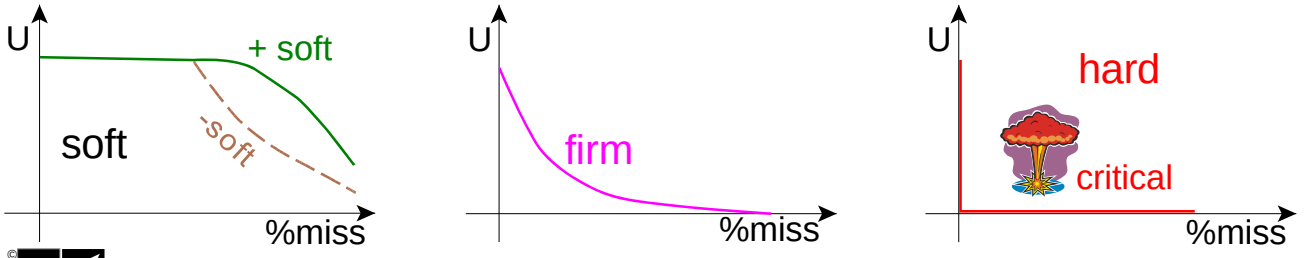
- **SCADA** means Supervision Control and Data Acquisition, which represents the market of the "control rooms" of industrial plants (power stations, petrochemistry, purification of water....); the time-constants extend from the millisecond to a few seconds.
- **telecom/multimedia** represent the various data transmissions, including voice and images, and their routing through the telecommunications networks; the time constraints are quite variable, from the simple e-mail (for which a delay of several minutes is more than acceptable) to the commutation of frames in a Gbps network.
- **electrical drives** are a field which covers industry (paper mills, metallurgy..) and the electric traction; electric time-constants are among the most severe ones: 1 to 10 milliseconds for the risetime of currents and a few µs for the protection of semiconductors.
- **MMI** means "Man Machine Interface"; the reaction times are on a human scale, i.e. some tenths of a second

Depending on the required performances, the solutions are of three types, with significant overlaps:
- for the **least severe** constraints, computer platforms and operating systems used for " **office**" applications are appropriate
- on the opposite, the **most severe** situations can only be handled by systems in wired or programmable logic (like **FPGA**)
- in the midrange, we find systems mixing **analog** and **digital** electronics using wired or **programmable logic**, small or powerful **microcomputers**/**microprocessors** running **operating systems** specially developed or the real-time operation.

Hard, *Firm* and *Soft* Real-Time

a rather fuzzy classification

- response time
  - soft, firm > 100ms (thermal, MMI, supervision)
  - hard < 100ms (electromechanics)
- utility of the results (U) if dead-line not met
  - soft: systems still works acceptably if significant % of missed deadlines
  - firm: degradation of performances (ex. voice)
  - hard: results no more useful if any missed deadline (ex.automatic pilot)
- criticity (risk estimation, safety, security)
  - soft => not critical
  - hard : sometimes critical (ex: ABS, avionics)



The distinction is sometimes difficult and fuzzy between the so-called " **hard**", "**firm**" and "**soft**" real-time. It is generally admitted that robots and electric drives are "hard" because of their small time-constants (< 100ms). Another criterion is the degradation of the "performance" of the system according to its slowness (tardiness) or the percentage of missed deadlines. In the three purely qualitative graphs above, the performance is expressed as the utility U of the produced results (for example the intelligibility of the voice) which can be evaluated by a performance index, generally a weighted (hence subjective) sum of error criteria.

- the "soft" system on the left figure tolerates a certain percentage of missed deadlines before its output becomes less usable. The degree of "softness" is of course highly dependent on the system.

- the central figure represents a "**firm**" system, because the utility U of the output decreases immediately from the first missed deadline.

- on the right, we have the "hard" system, where the system does not function anymore if the time constraints are not respected.

The system becomes **critical** if the consequences pose security or safety problems. The "soft" systems are certainly not critical. Therefore, the **critical systems are hard ones**.

The borders are fuzzy and often rest on probabilistic criteria:
- for a high probability of tardiness to what extent is the loss of performance acceptable?
- for a very rare occurrence of tardiness, what probability is considered as acceptable?

We enter here in the field of evaluation of criticality or the "acceptable industrial risk". Let us take the example of emergency braking on a car; the information in sensed at the pedal and transmitted via a real-time network to the processor responsible for the illumination of the rear red stop lights. The actual speed is 108 km/h (30m/s). Each millisecond of delay represents 30mm of distance for the driver of the car that follows. Which is the acceptable response time? The answer depends chiefly on another factor: the response time of the stop lights controller must be negligible compared to heating time of the bulb (200ms)+ the driver's reaction time (at least 300ms between the moment he sees the red light and the moment he begins to push on his braking pedal). Light-Emitting Diodes are appropriate for this application because they work instantaneously.

ELEC-H410 : Introduction

CONTENTS

- example: ignition of an internal combustion engine
- concept of real-time
- **specificities of embedded systems**

Embedded systems are

µP systems dedicated to a specific task

- within a *device*
  - peripheral : printer, scanner
  - telecom : router, mobile phone
  - audio-video : TV, Hi-Fi, camera
- in an *industrial process*
  - controller of physical values
    - p,T°,F,x,x',x'',I,V,P,f
  - sequencer of events (assembly line,...)
  - security/safety devices (railways signalling, ABS)
  - automotive : ignition, fuel injection,...

The term of *embedded system* has a connotation of "hidden" or rather discrete system.

- it is sometimes reduced to only one component: the microcontroller or a specialized integrated circuit (ASIC)
- it is often internal to a device or a process; in most of the cases, the user is unaware of its presence. Who knows (and cares) where and how many microprocessors are present in his car, radio or camera?

Compared to general purpose information systems, embedded systems are much more specific.

Embedded systems

have to meet specific constraints

- performance
  - objective, seen by the controlled process
  - **real-time**
- consumption, size
  - battery-powered instruments
  - portable instruments, satellites
- environment
  - physical: T^o, humidity, vibrations, dust, ...
  - electrical: EMI pollution, power quality
- price
  - always important, chiefly for larges series
  - less critical for space and military applications

Some constraints are also specific to embedded systems and differentiate them from other computer systems, in particular:

- contrarily to the systems where the performances can be subjective, the embedded systems must meet objective constraints related to the controlled process, going up to "hard" real-time

- many embedded systems are battery-powered portable devices, or have to function within a limited power budget (satellites are a good example)

- the environmental constraints to which they must answer are often severe; they are in particular subjected to
  - high temperatures (e.g. under the hood of a car)
  - high moisture (marine or tropical systems)
  - severe vibrations or accelerations (trains, planes....)
  - corrosive or abrasive dust (chemical, cement factories)

- the majority of the industrial environments also submitted to strong EMI (electromagnetic interferences) brought via the wires, or via capacitive or inductive couplings

- finally, like in all systems, money is a factor which can practically never be neglected

Embedded systems

require specific software and tools

- tasks
  - mono-task : single program
  - simple multitasking
    - no file => no file management
    - no OS
    - RTOS : Real-time OS
- storage
  - frequently in ROM (FLASH, E²PROM)
- criticality
  - real-time mandatory
  - error => security/safety consequences
- specific development tools
  - cross-compilers , cross debugger, real-time debug
  - emulators, logic analysers, ...
  - hardware/software co-development

Embedded systems are also specific in their software aspects.

Many embedded processors run a single program, in the absence of any operating system.

The small multitask systems are generally based upon a minimal operating system (the kernel), especially because there is no need to manage files; this operating system can itself be especially designed for real-time (RTOS Real Time Operating System).

The mass memory is small, to reduce cost, consumption and price. Frequently it is completely static, the discs being replaced by FLASH memories.

The development of the software for embedded systems requires specific tools
- because debugging must also occur in real time
- because of the necessity to avoid critical bugs
- because the development of the software is sometimes done in parallel wiht the development of a specific hardware (integrated circuits,..): this is called **co-development**

Embedded systems

run on specific processors

- general purpose microcomputers
  - large variety of applications due to a great flexibility on standard hardware
  - few powerful and expensive µP
    - x86 and successors
    - RISC
- embedded systems
  - large variety of applications
  - optimized systems dedicated to a task
  - large variety of processors
    - several 100
    - chiefly microcontrollers
    - System on Chip (SoC, MPSoC) and on-the-fly reconfigurable systems
    - cost 1 to 100$

In the general purpose microcomputers and the workstations, we currently find only two or three main families of processors (X86, SPARC.....).

On the other hand, in the embedded market, the optimization of performances, consumption and cost led to the creation of several hundreds of different microprocessors/microcontrollers, as well as ASICs.

The trend goes towards the Systems-on-Chip (SoC), a kind of super microcontrollers mixing programmed and programmable logic; they can be reconfigurable according to the application, even in the course of operation. They can be multiprocessor (MPSoC) to carry out the most severe processing (video (de)compression, real-time image analysis).

Embedded systems

have limited resources

- constraints
  - cost proportional to
    - Si area
    - size of case
  - consumption proportional to
    - number of transistors
    - frequency
    - V_{DD}^2
- limitation of resources
  - frequently some KB of memory
  - pins with multiplexed functions

The constraints of cost and consumption can bring a severe limitation of the number of resources. The world of microcontrollers is also populated with billions of small species with
- very few pins (a few tens, down to 6)
- a few tens of bytes of read-write memory
- a few KBytes of read-only memory.

In the majority of the cases, the pins can fulfill several functions (up to 3 or 4, but only one at a time).

Embedded systems

must have a long lifetime

- classical computers
  - new generation of µP / 18 months
  - lifetime: 3 to 5 years
- embedded systems
  - long life equipments
    - automotive 10+ years
    - railways, avionics, space: 20-30 years
  - industrial processors and components
    - extended temperature range
    - longer life; end-of-life (EOL) announcements
  - a guarantee of durability is increasingly difficult
    - likely to have to redesign for maintenance

In the field of computer systems, we assist to a continuous and rapid change; the lifetime of the hardware - and specially of the processors - is relatively short; many companies like banks or insurances change their equipment every three to five years.

This situation is unimaginable in most industrial applications; who would admit having to replace any processor of his car regularly because a new model has been pushed on the market?

In the industry, lifetimes from 10 to 30 years are quite frequent, in particular in the avionics and railways.

It is thus necessary for the manufacturers of embedded systems to rely on products
- with the longest possible period of manufacture
- for which there will be an announcement of the EOL (End-Of-Life) i.e.
  - the date from which a component will not be recommended any longer for new designs
  - the date until which customers will be able to still place an order with unlimited quantity
  - the date at which the manufacturing will be stopped

The tendency goes nevertheless towards the shortening of the lifetimes of the components. Offering a maintenance service during 30 years becomes a challenge. It can happen that a spare part is no longer available and has to be redesigned with new components to be functionally equivalent and geometrically compatible.

Conclusions

- industry needs
  - real-time
  - long lifetime
  - safe work in hard environments
  - efficiency
- => embedded systems
  - specific hardware (micro-controllers, FPGA,..)
  - specific real-time OS (RTOS)
- trends
  - decentralization (multi-processor)
  - interconnection via real-time networks