

Embedded System Design and Modeling

I. Introduction to Embedded Systems Design

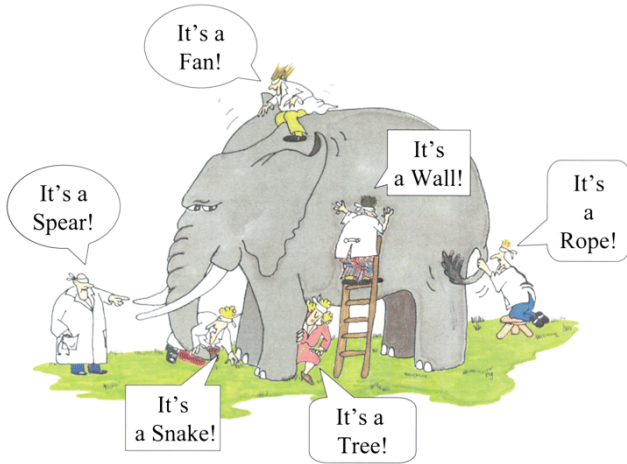
What are Embedded Systems?

- ▶ Embedded System (Marwedel 2011): Embedded systems are information processing systems embedded into enclosing products
- ▶ Cyber-Physical Systems (Lee & Seshia 2017): A CPS is an integration of computation with physical processes whose behavior is defined by both cyber and physical parts of the system
 - ▶ “cyber” means \approx “control” (from Greek)
- ▶ Key points:
 - ▶ there is a physical process to be controlled
 - ▶ there is some computational device who controls it
 - ▶ the processing is **close** to the physical process:
 - ▶ spatially: done right there (embedded)
 - ▶ behavioral: dedicated / specific to a particular process

What are Embedded Systems?

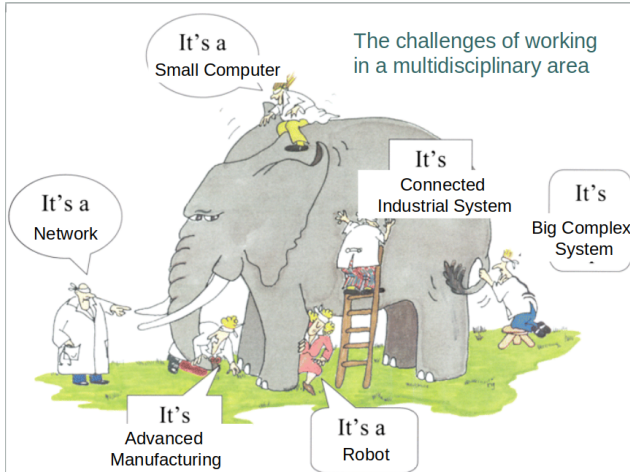
- ▶ Synonyms (more or less):
 - ▶ Embedded Systems
 - ▶ Internet of Things (IoT)
 - ▶ Industrial Internet
 - ▶ Systems of Systems
 - ▶ Industry 4.0
 - ▶ Internet of Everything (IoE)
 - ▶ Smart
- ≈ Cyber-Physical Systems

What are Embedded Systems?



► Image from Lee&Seshia 2017

What are Embedded Systems?



► Image from Lee&Seshia 2017

Found everywhere

- ▶ Embedded systems are everywhere:
 - ▶ Automotive (Transportation industry)
 - ▶ Telecommunications
 - ▶ Medicine
 - ▶ Consumer electronics
 - ▶ ...

Common characteristics

- ▶ Embedded systems share common characteristics:
 - ▶ must be **dependable**
 - ▶ reliability: probability that a system will not fail
 - ▶ maintainability: probability that a failed system can be repaired
 - ▶ safety: does not cause any harm even in worst-case conditions
 - ▶ security: allows authentication and confidentiality of data
 - ▶ must be **efficient**
 - ▶ low power consumption
 - ▶ low weight
 - ▶ low cost
 - ▶ no unnecessary resources used

Common characteristics

- ▶ Embedded systems share common characteristics:
 - ▶ must satisfy **strict timing constraints**
 - ▶ most embedded systems operate in real-time
 - ▶ must be **fault-tolerant**

Embedded systems vs PC

- ▶ Aren't embedded systems just "small PC's"? No.

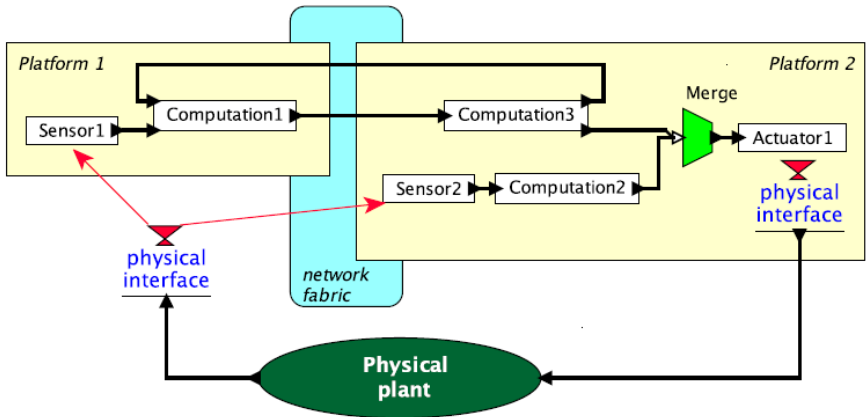
	Embedded	PC-like
Architectures	Frequently heterogeneous very compact	Mostly homogeneous not compact (x86 etc)
x86 compatibility	Less relevant	Very relevant
Architecture fixed?	Sometimes not	Yes
Model of computation (MoCs)	C+multiple models (data flow, discrete events, ...)	Mostly von Neumann (C, C++, Java)
Optim. objectives	Multiple (energy, size, ...)	Average performance dominates
Real-time relevant	Yes, very!	Hardly
Applications	Several concurrent apps.	Mostly single application
Apps. known at design time	Most, if not all	Only some (e.g. WORD)

Figure 1: Embedded Systems vs PC

- ▶ Image from Marwedel 2011

Structure of an embedded system

- Typical structure of an embedded system (CPS)



- Image from Lee&Seshia 2017

Structure of an embedded system

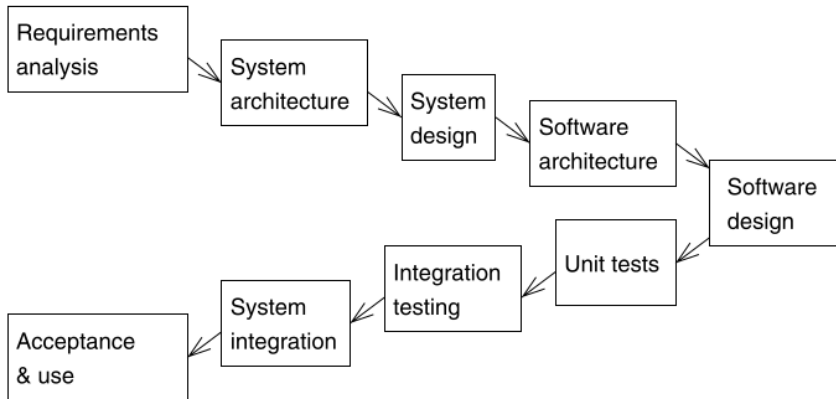
- ▶ Main components:
 - ▶ the physical process (known as the “plant”)
 - ▶ sensors: acquire information from the process
 - ▶ actuators: act on the process
 - ▶ computation: may be split between different devices
 - ▶ communications: between separate devices

The design process

- ▶ Iterative:
 - ▶ **Modeling:** “the process of gaining a deeper understanding of a system through imitation. It specifies what a system does.”
 - ▶ **Design:** “the structured creation of artifacts. It specifies how a system does what it does.”
 - ▶ **Analysis:** “the process of gaining a deeper understanding of a system through dissection. It specifies why a system does what it does.”
 - ▶ ... and iterate again.

The V-Cycle

- Common System Development Cycle in automotive industry (from a SW point of view):



- Image from Marwedel 2011

The V-Cycle

Steps:

- ▶ Requirements specifications
- ▶ System Architecture & Design
 - ▶ part of the Design and Modelling is done here
- ▶ SW Architecture & Design
 - ▶ part of the Design and Modelling is done here
- ▶ Implementation
- ▶ Tests & Validation
 - ▶ all tests are done against the corresponding documents from the other branch

Other processes

- ▶ For System & Software: Waterfall, Agile
- ▶ For Hardware: Gajski's Y-chart

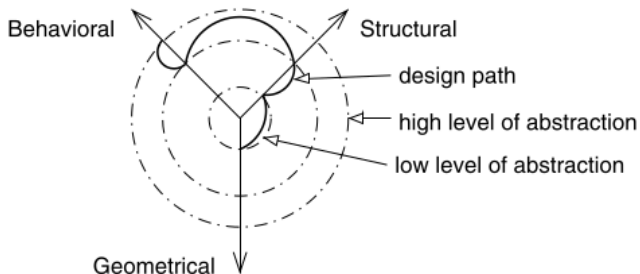


Figure 2: Gajski Y Chart and design path

- ▶ Image from Marwedel 2011