Computational Models for Embedded Systems

Vescan Andreea, PHD, Assoc. Prof.

Faculty of Mathematics and Computer Science
Babeș-Bolyai University
Cluj-Napoca

2025-2026

Lecture 1: Introduction to Computational Models for Embedded Systems

Faculty of Mathematics and Computer Science
Babeș-Bolyai University

“Tell me and I forget, teach me and I may remember, involve me and I learn.”

(Benjamin Franklin)

Outline

- **Embedded Systems**
  - **What**
  - Where
  - Promise & Future
- Model. Model of computation
- Questions

What are Embedded Systems (ES)?

- ESs - information processing systems that interact with physical processes and are embedded into a larger product.
  - Physical constraints: time, cost, power.
  - Software Engineering in Real Time.
  - Multiple Stimulus/Response loops.
- ES - “Any device that includes a programmable computer but is not itself intended to be a general-purpose computer.” [Marilyn Wolf]

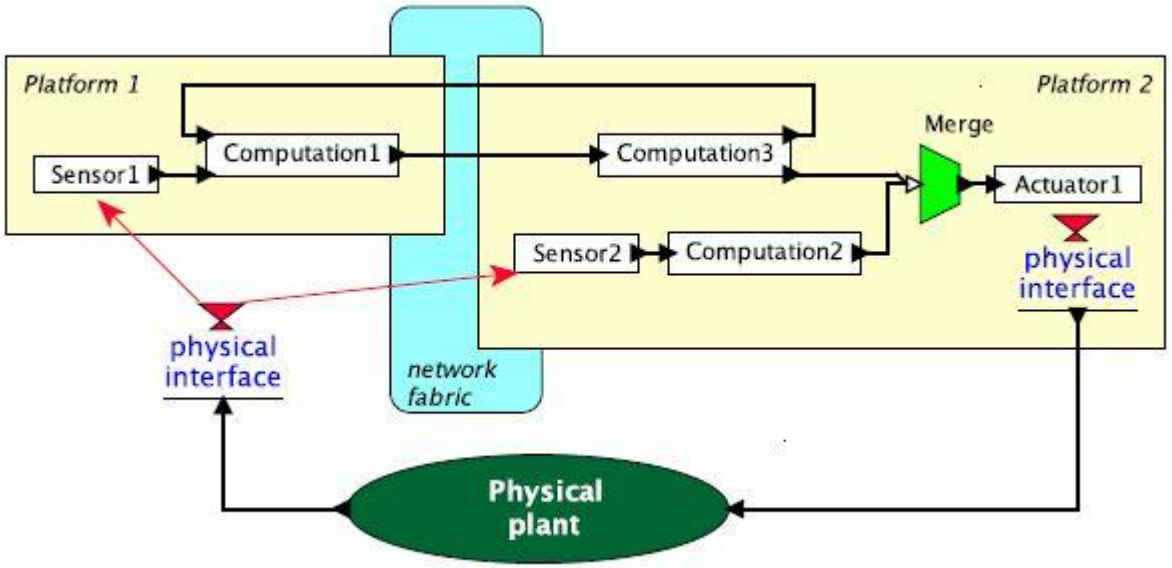
- **Example 1 - Heart surgery – requires stopping the heart**

  Operate on a beating heart?
  - Surgical tools – robotically controlled – move with the motion of the heart
  - Stereoscopic video system – video illusion of a still heart to the surgeon

- **Example 2 - Traffic lights and cars cooperate**

Not having to stop at a red light (except actual cross traffic)?
- Detects cars on the road
- The cars to cooperate.

Cyber-physical system



Example structure of a cyber-physical system [2]

Outline

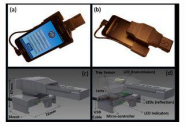
- **Embedded Systems**
  - What
  - **Where**
  - Promise & Future
- Model. Model of computation
- Questions

Where Embedded Systems (ES) ?

- Automotive
- Aerospace
- Mobile
- Medical Instruments
- Mobile medical device
- Robotics
- Process control
- Sensor Nets
- Consumer Electronics
- Multimedia
- E-Business

- Emerging direction: cell phone based medical devices for affordable healthcare
  - CellScope- Telemicroscopy project at Berkeley University.
    - http://thefutureofthings.com/3422-the-birth-of-the-cell-phone-microscope/

  - Cell-phone based blood testing device developed at UCLA (University of California, Los Angeles).
    - https://newsroom.ucla.edu/releases/ucla-researchers-create-smartphone-based-device-that-reads-medical-diagnostic-tests-quickly-and-accurately



Outline

- **Embedded Systems**
  - What
  - Where
  - **Promise & Future**
- Model. Model of computation
- Questions

Promise and Future of ES

- The Promise of Embedded Systems (video from 2012)
  - 2012 – (6minutes)
    - http://www.youtube.com/watch?v=jZkHpNnXLB0
  - 2014 – (3-4 minutes)
    - https://www.youtube.com/watch?v=XyIvSIY0MTM
  - 2016- (4-5 minutes)
    - https://www.youtube.com/watch?v=CPH1sAY8Vhg
  - Watch your day in 2020
    - https://www.youtube.com/watch?v=jFir46J0RsE&ab_channel=MaraRocha

- Future of Embedded Systems
  - Ubiquitous and pervasive computing:
    - Information anytime, anywhere; building ambient intelligence into our environment; internet of things:
  - Wearable computers
  - "Smart Labels" on consumer products
  - Intelligent buildings
  - Environmental Monitoring
  - Traffic control and communicating automobiles

- Embedded systems provide the basic technology.

Sun's Version of the Wired House
- Will people adopt this other than as a toy?
- Will the same people who can't set time on a VCR be able to debug their house?

- Life in 2050? (6 min.)
- https://www.youtube.com/watch?v=hT9GpCJj6nw&ab_channel=PorViver
- Use
  - Menti
  - Life in 2050? What do YOU think?

Outline

- Embedded Systems
  - What
  - Where
  - Promise & Future
- **Model. Model of computation**
- Questions

Model

- ***Activity: What is a model?***
  - open question

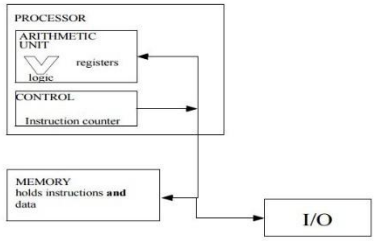
- A model is *a* ***simplification of another entity***, which can be a physical thing or another model. The model contains exactly those *characteristics and properties* of the modeled ***entity*** which are ***relevant for a given task***. A model is minimal with respect to a task, if it does not contain any other characteristics than those relevant for the task.

- **e.g. Tower Bridge London**

Models and Abstractions

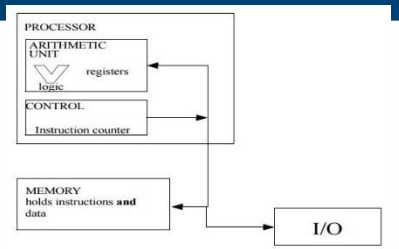
- Activities usually start with informal specification
- Models and Abstractions soon follows
  - Abstraction enables decomposition of systems into simpler sub-systems (chess-components=pieces, composition rules=playing board, movement rules)
  - Models proved structure on which analysis and optimization are possible.
- Two types of modeling:
  - system structure
  - system behavior
    - behavior is externally visible events based on internal interactions of abstract components
    - properties are constraints met by all behaviors of a system.

Model of computation (1)



- A **model of computation** governs the interaction of components in a design.
- In the classical Von Neumann model of computation
  - the components are statements that transform the state of a data store.
  - a report on the stored program concept – on EDVAC (Electronic Discrete Variable Automatic Computer)
  - The basic structure proposed:
    - a memory, containing instructions and data;
    - a processing unit, for performing arithmetic and logical operations;
    - a control unit, for interpreting instructions.
- In embedded software design
  - components are concurrent processes or threads, typically scheduled by a priority-driven real-time operating system.

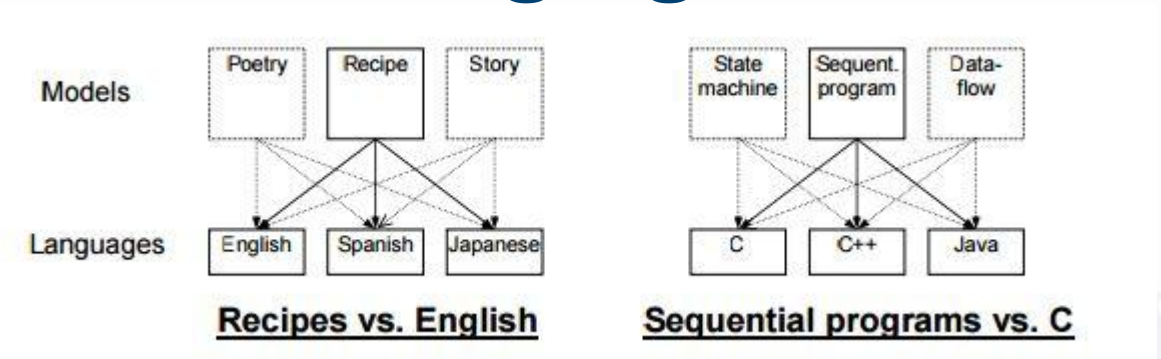
Model of computation (2)



- A design is represented as a set of components, which can be considered as isolated monolithic modules (often called processes or tasks), interacting with each other and with the environment. The **model of computation** defines the **behavior and interaction mechanisms of these modules**.

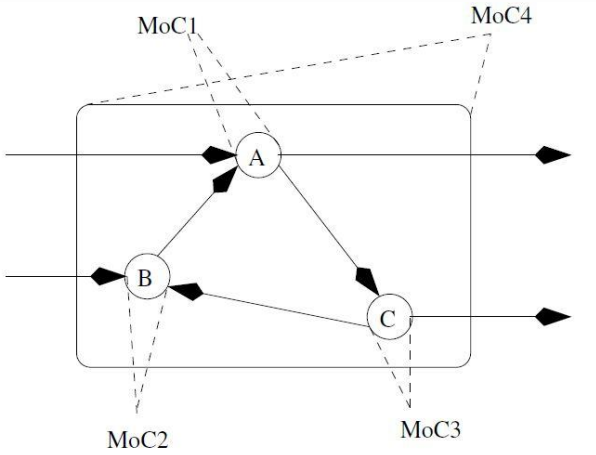
- Models of computation usually refer to:
  - how each module (process or task) performs **internal computation**;
  - how the modules **transfer information** between them;
  - how they relate in terms of **concurrency**.
- Remark: Some models of computation do not refer to aspects related to the internal computation of the modules, but only to module interaction and concurrency.
- The main aspects we are interested in:
  - Concurrency
  - Communication & Synchronization
  - Time
  - Hierarchy

Models vs. Languages



- Computation models describe system behavior
  - Conceptual notion, e.g., recipe, sequential program
- Languages capture models
  - Concrete form, e.g., English, C
- Variety of languages can capture one model
  - E.g., sequential program model C,C++, Java
- One language can capture variety of models
  - E.g., C++ → sequential program model, object-oriented model, state machine model
- Certain languages better at capturing certain models.

Heterogeneous Models



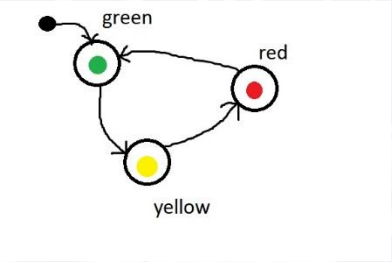
ES are often Heterogeneous:

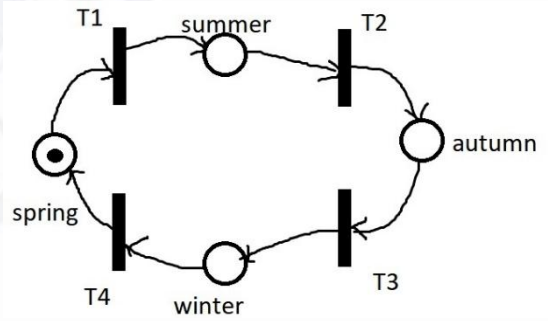
- A system consists of different parts.
  - E.g. data flow and control flow dominated parts
- Different tasks during the development process need the inclusion/exclusion of certain information
- (specification/verification/simulation/synthesis)
- Different communities have got acquainted with different formalisms and tools.
- Unifying notations/formalisms is too difficult (if not impossible). Researchers have started to link and integrate different languages.

(Common) Models of Computation

ES are often Heterogeneous:

- Some of the models of computation commonly used to describe embedded systems:
  - (Synchronous) Finite State Machines
  - GALS Models (Globally asynchronous locally synchronous)
  - Dataflow Models
  - Petri Nets
  - Discrete Event
  - Timed Automata





References
Sources

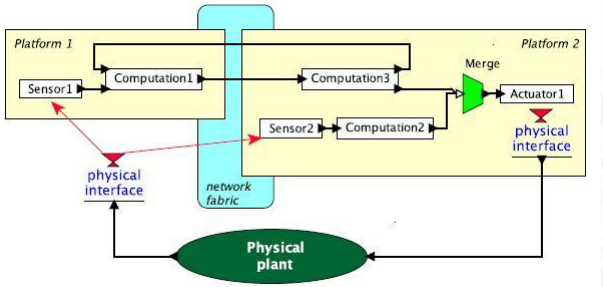
[1] Marilyn Wolf, *Computers as Components: Principles of Embedded Computer Systems Design*, Publisher: Elsevier/Morgan Kaufmann, Year: 2017, ISBN: 9780128053874,0128053879

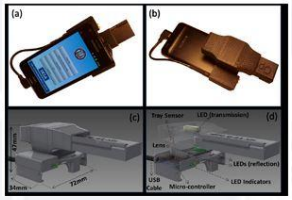
[2] Lee and Seshia, Introduction to Embedded Systems - A Cyber-Physical Systems Approach, 2017

https://ptolemy.berkeley.edu/books/leeseshia/

CMES – Today Bring it All Together

Embedded svstems: what + where + Promise &Future





Robots will be everywhere

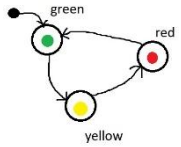
Model. Model of computation

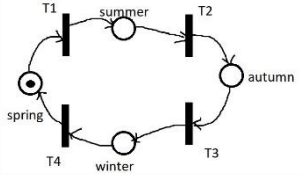
- A model of computation

  governs the interaction of components in a design.
- In embedded software design

  components are concurrent processes or threads, typically scheduled by a priority-driven real-time operating system.

- Models of computation usually refer to:
  - how each module (process or task) performs **internal computation**;
  - how the modules **transfer information** between them;
  - how they relate in terms of **concurrency**.





Thank You For Your Attention!

- ExitTicket
- Mentimeter
  - menti.com
  - Code: ?

Next Week

- Lecture 2
  - Model checking