Computing Needs Time

Edward Lee, UC Berkeley (2009)

Presented by Ricardo Severino (CISTER/ISEP)

OVERVIEW

- CPS and current status of embedded systems
- Requirements to enable CPS
- Timing and today's computing abstractions
- Interesting aphorisms
- Solutions
- Final Remarks

CPS – CYBER-PHYSICAL SYSTEMS

- Today uP are embedded in systems such as:
  - Cars, industrial robots, toys, games, medical devices, etc.
  - Tomorrow, in everyday life objects.
- There is a tight interaction with a physical process by sensing and actuating actions
- “Orchestration of networked computational resources with physical processes”.

ANOTHER CPS EXAMPLE (PRINTING PRESS)

- High speed and high precision machine
  - Speed (1 inch/ms)
  - Ink precision (0.01 inch)
    - Time accuracy 10us
  - Hundreds of controllers
  - Thousands of sensors
- Communications
  - Ethernet (1588 time-sync protocol)
- What if a failure occurs (paper jam)
  - Power down is not na option
- Must have
  - Models of the physical process as a part of the software
  - Precise time control of the network


Bosch-Rexroth

CPS – Cyber-Physical Systems

- Today uP are embedded in systems such as:
  - Cars, industrial robots, toys, games, medical devices, etc.
  - Tomorrow, in everyday life objects.
- There is a tight interaction with a physical process by sensing and actuating actions
- “Orchestration of networked computational resources with physical processes”.
- They are becoming networked and intelligent, however sometimes at the cost of dependability.
- Increasing lack of confidence in technology.

What?! Houston, we have a problem!

The "Low-Fi" era

- Computers now
  - integrate media such as video and audio
  - In handheld platforms – Sense physical dynamics and control physical devices. "Baah, not works good."
- However, they don't do it very well, do they?
  - Video quality we get in the internet can be sometimes worse than television broadcast in the 1950's.
  - Audio quality in many telephone connections – voice is incomprehensible. First digital telephony systems in the 1960's were better!
  - In set-top boxes, designers discard many innovation of the last 30 years of computing (no OS, no high-level programming languages, no memory management, nor abstractions that hide temporal properties on their interfaces. (Time is not irrelevant).

What is wrong with this picture?!

CPS/Traditional Systems

- Traditional Embedded systems
  - Embedded software on small computers
  - Concerned about optimizing the use of limited resources.
- CPS
  - Computation and networking integrated with physical processes
  - Concerned about managing dynamics, time and concurrency.
  - All components are tightly coupled, so they must be jointly designed.

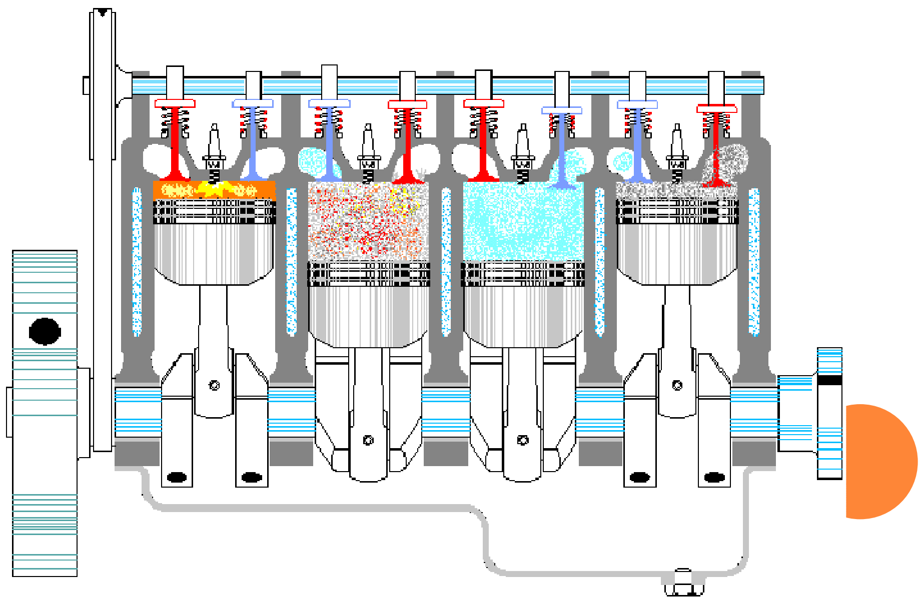
Systems Reliability (1)

- Embedded systems have been held to a higher reliability standard than general-purpose computing (we do not expect TV to crash and reboot)
  - Computer means higher reliability and efficiency.
  - Design analysis - predictability
  - Test under different operating conditions - repeatability
- CPS – expect even higher reliability (traffic control, automotive safety…)
  - Simplest C program not predictable nor repeatable because

"***The design does not express aspects of the behavior that are essential to the system***"

Systems Reliability (2)

- Exact match with C semantics and still fails to deliver behavior nedded by the system!
  - Timing deadlines – NOT in C semantics
- Failure of abstraction
  - A predictable and repeatable component fails in the dimensions that matter.



But still…

- We can move outside C and use OS primitives for I/O
  - But then we get non-deterministic behavior
- Must use semaphores, locks, priorities, inter-process communication
  - harder to test and can produce brittle systems.
  - A small change in the design can cause major problems in the system.

THE AIRCRAFT

- Boeing 777 aircraft (fly-by-wire)
  - A slight change to the harware may affect timing and require re-certification

Software certification:
1 billion US dollars

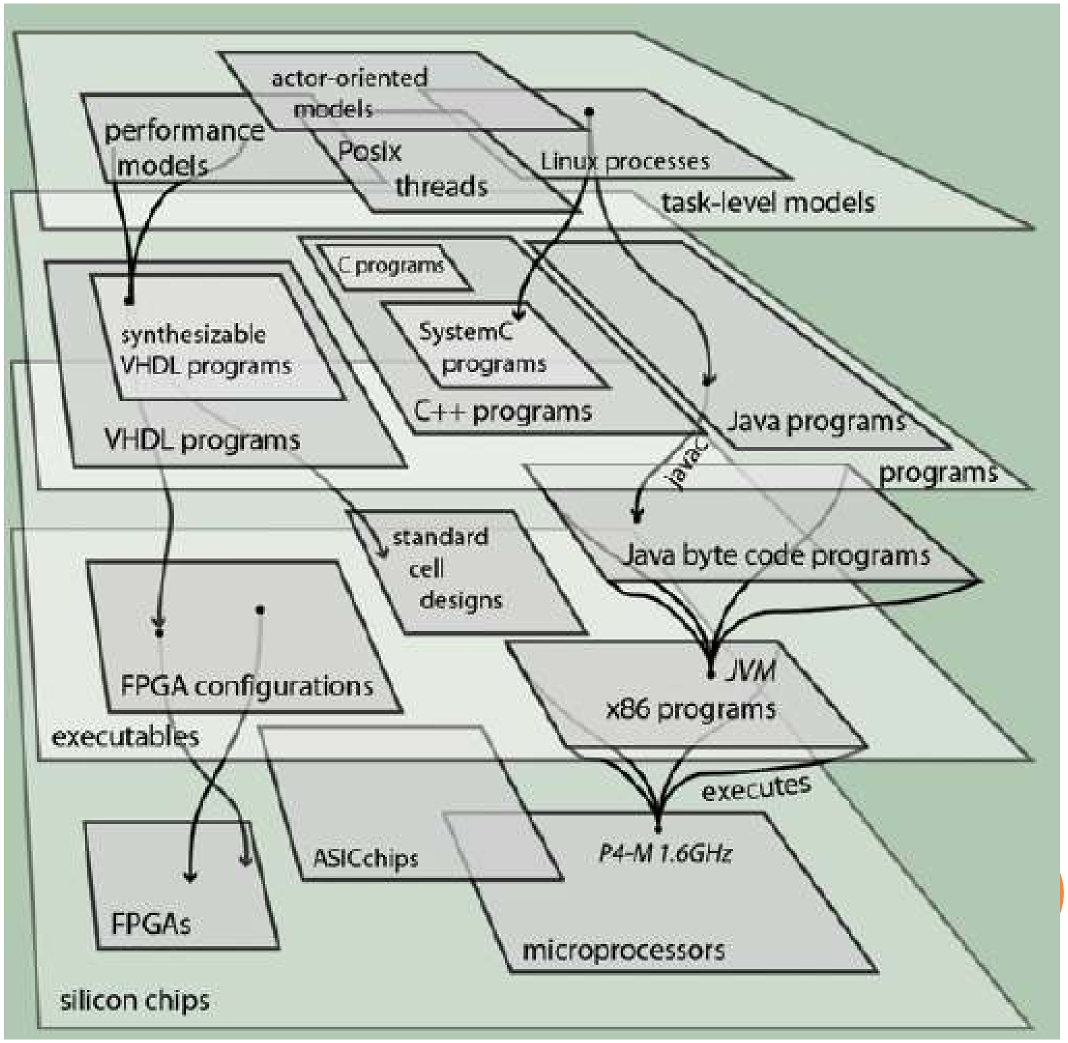
Aircraft model life span:
50 years

50 years stockpile hardware components that execute the software

Relying on uProcessors designed in the 90's in 2030

ABSTRACTIONS…

- "Today's computing and networking technologies unnecessarily impede the progress towards CPS applications."



OK, BUT WHY?!

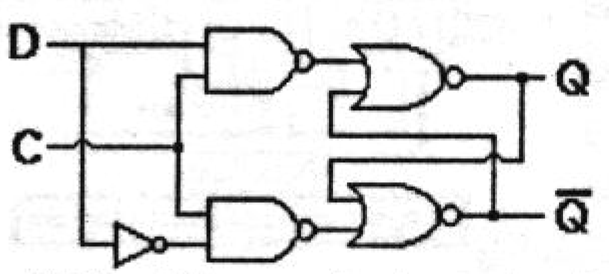
- Nearly every abstraction has failed!
- The ISA - ISA users care about timing properties ISA cannot express. (WCET in modern processors)
- The programming language – no widely used programming language expresses timing properties. Timing is an accident of implementation.
- A realtime operating system - fails if the timing of the underlying platform is not repeatable or execution times cannot be determined.
- The network - most standard networks provide no timing guarantees and fail to provide an appropriate abstraction.

WHERE DOES THIS PROBLEM COMES FROM?

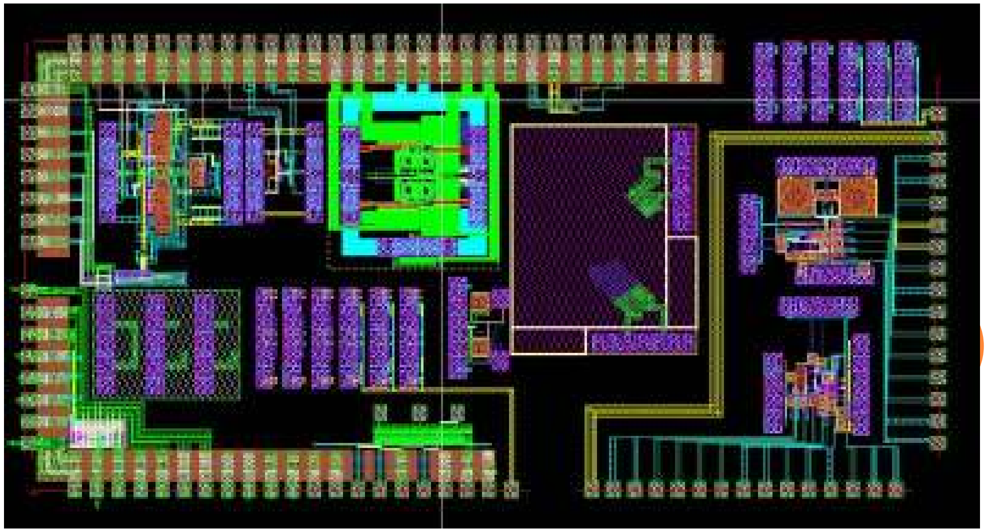
- Is this problem intrinsic to technology?
  - Electronics technology delivers highly repeatable and precise timing!
  - Yet, overlaying abstractions discard it…

    BY CHOICE!

    There is a call for a fundamental change in the core abstractions of computing!



D Flip–Flop equivalent circuit




YOU'RE DOING IT WRONG

Interesting Aphorisms (1)

- "Time is a non-functional property"
  - What is the function of a program?
    - Turing-Church: finite composition of functions whose domain and codomain are a set of sequences of bits.
    - CPS: Defined by its effect in the physical world. Here domain and codomain of the function are not sequences of bits.
- Why insisting in a wrong definition of "function"?
  - A program is a sequence of input/output events for many system designers (OS, web servers, comm. Protocols).
  - Elevate this view to the application programmer and augmented with temporal dynamics.

INTERESTING APHORISMS (2)

- “Real-Time is a quality of service problem”
  - Everybody wants quality!
  - In general-purpose computing: Execution time = performance.
  - In embedded systems: Less time is not better that more time! (e.g. firing engine spark plugs earlier will not make it more performant)
- CPS requires repeatable timing behaviour far more than optimized performance.
- QoS problems are Timing Precision and Variability.

SOLUTIONS (1)

- Computer Architecture
  - ISA provides different performance without losing compatibility with existing harware. (1960's – IBM360)
  - Today's ISAs hide most temporal properties of underlaying hardware.
- Extend ISA with timing properties.
- "Achieving timing precision is easy if system designers are willing to forgo performance";
  - although cache memories may introduce unacceptable timing variability, cost-effective system design cannot do without memory hierarchy.
  - Challenge: provide memory hierarchy with repeatable timing.
  - Similar challenges apply to pipelining, bus architectures, and I/O mechanisms.

SOLUTIONS (2)

- Programming languages
  - Abstraction layer above ISA
  - We need to reflect the underlaying temporal properties in the language semantics
  - Annotate programs can be a faster solution than creting a new language.
  - Another idea is to integrate other domain specific languages with temporal semantics such as Simulink or LabVIEW, into engineering processes.

SOLUTIONS (3)

- Software components
  - Data abstraction, object orientation, and component libraries made it easier to design large complex systems.
  - Most of these component tech. Do not export temporal properties in the APIs.
    - Could provide na interesting alternative to real-time programming languages.
  - New coordenation languages, with components based in (Java/c++) may be more likely to gain acceptance.

Solutions (4)

- Formal Methods
  - User mathematical models to infer and prove properties of systems
- Several approaches that handle temporal dynamics
  - Temporal logics
  - Process algebras
  - Timed-automata
- However, properties not formally specified cannot be formally verified. (e.g. software timing behaviour, is not expressed in software, must be separately specified)
  - Breaks connection with implementation.
- All depends on
  - progress on programming languages
  - Scalability to realistic systems

SOLUTIONS (5)

- Networking
  - Timing behaviour is viewed as QoS problem
  - designers of time-sensitive applications on general-purpose networks (such as voice over IP) struggle with inadequate control over network behavior.
- Meanwhile, FlexRay and TTA (time-triggered architecture) emerged to provide timeliness as correctness property.
- Introducing timing into networks as a semantic property rather than a QoS problem will lead to an explosion of new time-sensitive applications

Final Remarks

- To realize the potential of CPS core abstractions of computing must be rethought.
- Semantic models must reflect the properties of interest of the physical processes.
- Timing properties must become a correctness criteria and not a QoS measure.
- Timing in programs and networks must be repeatable and predictable as technologically feasible at reasonable cost.

THANK YOU FOR YOUR INTEREST!

Ricardo Severino

<rars@isep.ipp.pt>