A Story…

Fly-by-wire technology controlled by software.

Success?

They have to purchase and store microprocessors for at least 50 years production and maintenance…

Why?

Apparently, the software does not specify the behaviour that has been validated and certified!
**What is PRET?**

Timing is not part of the software semantics

Correct execution of programs (e.g., in C, C++, C#, Java, Scala, Haskell, OCaml) has nothing to do with how long time things takes to execute.

**Traditional Approach**

Programmer Model  \(\rightarrow\) Timing Dependent on the Hardware Platform

**Our Objective**

Make time an abstraction within the programming model  \(\rightarrow\) Timing is independent of the hardware platform (within certain constraints)

---

**What is Precision Timed (PRET) Infrastructure?**

A vision of making time first class citizen in both software and hardware.

**PRET Infrastructure**

- PRET Language (Language with timing semantics)
- PRET Compiler (Timing aware compilation)
- PRET Machine (Computer Architecture)
What do mean by precision, predictable, and repeatable timing?

Focus on cyber-physical systems with real-time constraints

- **Hard task**
  - Missed deadline: Catastrophic consequence
- **Firm task**
  - Missed deadline: Result is useless, but causes no damage
- **Soft task**
  - Missed deadline: Result has still some utility

**Predictable timing**
- Guarantee correctness (WCET)
- Early miss detection
- Immediate miss detection
- Late miss detection

**Precision of timing**
- Enable accuracy in nano seconds

**Repeatable timing**
- Same platform: Testability
- Changing platform: Portability

Processor frequency

**Task** (clock cycles)

Deadline

Time (measured in e.g., ns)

---

Languages with timing semantics

**Modeling Languages**
- Simulink/Stateflow (Mathworks)
- Modelica (Modelica Associations)
- Ptolemy II (Eker et al., 2003)
- Modelyze (Broman and Siek, 2012)

**Programming Languages**
- Real-time Concurrent C (Gehani and Ramamritham, 1991)
- PRET-C (Andalam et al., 2009)

**Assembly Languages**
- The assembly languages for today's processors lack the notion of time
Rethink the ISA
Timing has to be a correctness property not only a performance (quality) property

PRET Machine
• Repeatable and predictable execution time
• Repeatable memory access time
• Timing instructions for handling missed deadline detection

PRET Infrastructure

|---------------------|-------------------------------|---------------------------------|-------------------------------|---------------------------------|---------------------------------|

Programming Languages

Semantic gap between timed high level modeling languages and PRET ISA

Assembly Languages

PRET ISA
Can we just compile directly down to PTARM?

Lots of redundant work…

PRET Infrastructure

Modeling Languages
- Simulink/Stateflow (Mathworks)
- Modelica (Modelica Associations)
- Ptolemy II (Eker et al., 2003)
- Modelyze (Broman and Siek, 2012)

Programming Languages

Assembly Languages

PRET ISA

PRETIL vision

Modeling Languages
- Simulink/Stateflow (Mathworks)
- Modelica (Modelica Associations)
- Ptolemy II (Eker et al., 2003)
- Modelyze (Broman and Siek, 2012)

Programming Languages

- C extended with high-level timing constructs.
- Can be seen both as an intermediate and programming language

E machine (Henzinger, and Kirsch, 2007)

ptC

- Expose timing constructs
- Abstracting away memory the hierarchy (scratchpad, DRAM etc.)

PRETIL

Assembly Languages

PRET ISA
**PRETIL vision**

### Modeling Languages
- **Simulink/Stateflow** (Mathworks)
- **Modelica** (Modelica Associations)
- **Ptolemy II** (Eker et al., 2003)
- **Giotto** (Henzinger, Horowitz, and Kirsch, 2003)
- **Modelyze** (Broman and Sick, 2012)

### Programming Languages
- **ptC**
- **E machine** (Henzinger and Kirsch, 2007)

### Assembly Languages
- **Flexible PRET Machines**
- **PRET ISA**

### Languages
- **Assembly Languages**
- **Programming Languages**
- **Modeling Languages**

**PRETIL**
- C extended with high-level timing constructs.
- Can be seen both as an intermediate and programming language.

**Rewrites**
- High-level languages with Timing semantics
- Intermediate language with Timing semantics
- WCET Analysis and WCET-aware Compilation
- WCET-Aware Scratchpad allocation

**Precision Clock Synchronization**
- Flexible PRET Machines
- PRET ISA