CHESS
CENTER FOR HYBRID & EMBEDDED SOFTWARE SYSTEMS

COLLEGE OF ENGINEERING, UC BERKELEY

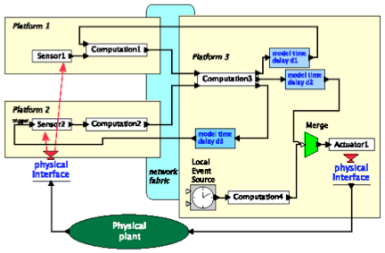
From PTIDES to PtidyOS: Programming Distributed Real-Time Embedded Systems

John Eidson
Jeff C. Jensen
Edward A. Lee
Slobodan Matic
Jia Zou

http://chess.eecs.berkeley.edu/

This work was supported in part by the Center for Hybrid and Embedded Software Systems (CHESS) at UC Berkeley, which receives support from the National Science Foundation (NSF awards #0720882 (CSR-EHS: PRET), #1035672 (CPS: PTIDES), and #0931843 (ActionWebs)), the U. S. Army Research Office (ARO #W911NF-07-2-0019), the U. S. Air Force Office of Scientific Research (MURI #FA9550-06-0312), the Air Force Research Lab (AFRL), the Multiscale Systems Center (MuSyC). The authors acknowledge the support of the Multiscale Systems Center, one of six research centers funded under the Focus Center Research Program, a Semiconductor Research Corporation program. one of six research centers funded under the Focus Center Research Program, a Semiconductor Research Corporation program, and the following companies: Bosch, National Instruments, Thales, and Toyota.

Distributed Real-Time System Implementation



Programming Model

Programming **T**emporally **I**ntegrated **D**istributed **E**mbedded **S**ystems

Based on Discrete-Event semantics

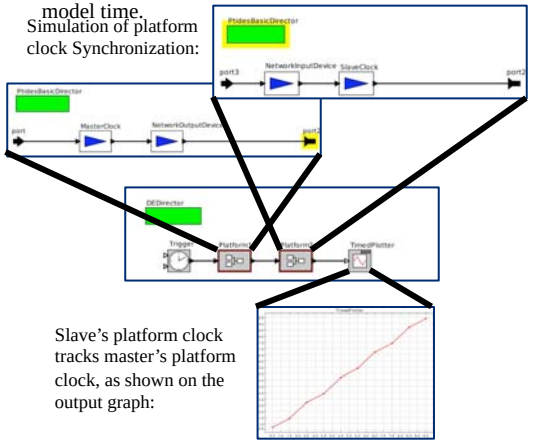
Event processing in time-stamp order

Clock Synchronization Simulation

Notions of Time:

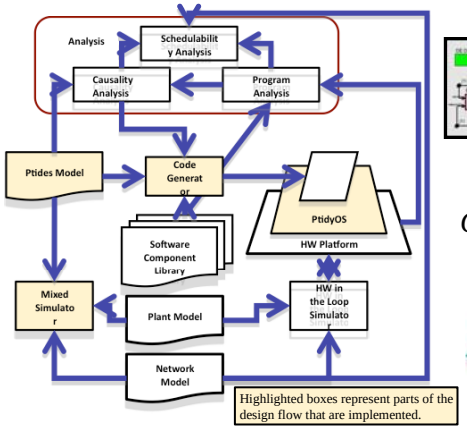
- Top level simulate real world (oracle) physical time.
- Each platform contains a clock to keep track its notion of platform physical time.
- Each Ptides director has a notion of model time.

Simulation of platform clock Synchronization:

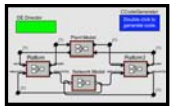


Slave's platform clock tracks master's platform clock, as shown on the output graph:

Ptides Design Flow



Highlighted boxes represent parts of the design flow that are implemented.



Code Generation

Ptides Simulator

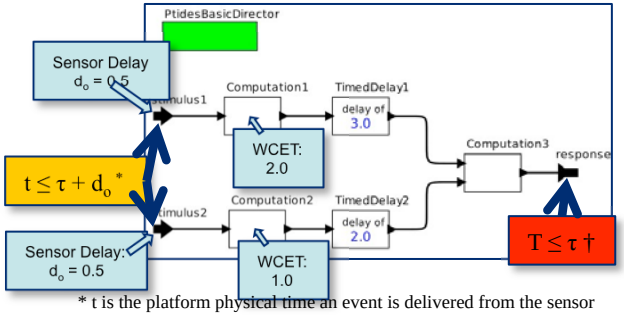
Simulate functional and real-time properties.

Function simulation :

- Same denotational semantics as DE simulation, i.e., Same functional outputs whether Ptides or DE Director is used.
- Performs safe-to-process analysis instead of DE simulation
- Currently support 2 Ptides execution strategies: Basic, PreemptiveEDF (combines Ptides with Earliest-Deadline-First).

Real-time simulation:

- Users annotate WCET and sensor delays.
- Check for deadline misses at actuators.



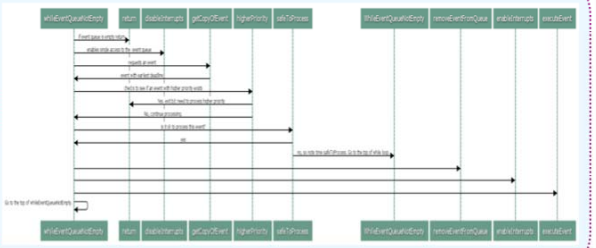
* t is the platform physical time an event is delivered from the sensor to the rest of the platform. τ is the timestamp of the sensor event.

† t is the platform physical time an actuation event is delivered to the actuator. τ is the timestamp of the actuation event.

PtidyOS

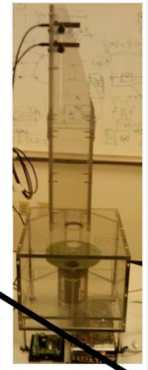
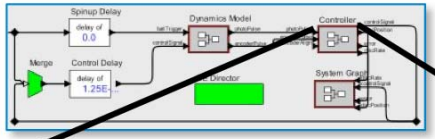
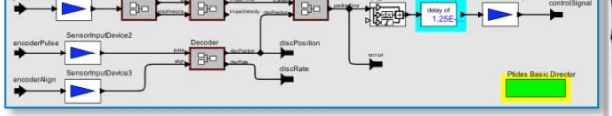
- Is a library linked against application C code.
- Implements Ptides semantics
- Local execution strategy includes a safe to process analysis and resource scheduling layer.
- Uses a Single Stack
- Steps away from threading model
- To ensure event of highest priority is always processed first, interrupts play an important role:
  - Event processing is done within Interrupt service routines
  - Reentrant interrupts
- No dynamic memory allocation.

Event Processing Algorithm Sequence Diagram



***The use of a single stack and interrupts for mutual exclusion was done in TinyOS. PtidyOS uses these concepts and introduces time semantics.**

Application: The Tunneling Ball Device

- Cyber-physical system with real-time constraints
- Event-driven signals (motor encoding, object detection)
- Parallels an automotive engine control unit (ECU)
- Timed semantics ensure:
  - Deterministic control delay
  - Data associated with the same physical time are processed together

- Balls drop above a rapidly spinning disc
- Device must adjust disc to allow ball to pass through a small hole, without stopping disc