Model-Based Development of Deterministic, Event-Driven, Real-Time Distributed Systems

Patricia Derler, John Eidson, Edward Lee, Slobodan Matic, Michael Zimmer

MBDEFP Workshop
October 13th, 2011
Taipei, Taiwan
Goals

Platform independent model of functional and timing behavior

Code Generation

Simulation

Same I/O behavior w.r.t. value and timing

XMOS development board with 4 XCores.

Renesas 7216 Demonstration Kit
Goals

Platform independent model of functional and timing behavior

Code Generation

Simulated

XMOS
Predictable timing
Multiple cores
No analog I/O
No FPU
No hardware clock

Renesas
PHY chip for accurate timestamping of inputs,
Analog I/O

Same I/O behavior w.r.t. value and timing

XMOS development board with 4 XCores.

Renesas 7216 Demonstration Kit

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
Goals

Model of functional and timing behavior

Code Generation

Simulation

Same I/O behavior w.r.t. value and timing
CPS
- an example
PTIDES

Event-triggered programming model

Event = Timestamp (model time) Value

Process causally related events in timestamp order
Produce event with timestamp = physical time

Perform actuation when timestamp = physical time
Model time delays

Explicitly modeled delay between sensing and actuating

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
Consume event only if no other event with smaller timestamp can arrive later during execution

= Safe-to-process analysis
PTIDES ... during an execution

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
PTIDES ... during an execution

Select one event out of all safe events

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
Generated Code

PtidyOS

• I/O handling
  – Interrupts, event mechanisms, time stamping, …

• scheduling of actors
  – Event queue management
  – Safe-to-process analysis
  – Scheduling, e.g. EDF, RM, …

• context switching

Actor code

• platform independent code for functionality and timing specifications

• platform dependent code for interfacing with the OS and the hardware.
Modeling in Ptolemy II

Model of Computation (MoC): actor scheduling, safe-to-process analysis

Functionality for interfacing with environment (Plant, Network) which is defined in another MoC

Ptides Director

Sensor1 → model time delay d4 → Computation1 → NI1 → Actuator

Timing

Functionality
Every model component has a code generation helper and a code template.
Code Generation in Ptolemy II

PtidyOS: actor scheduling, safe-to-process analysis

Actor specific functionality

Parse template code and fill in model specific parameters

Code Generation Helper

Code Template

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
Demo – Printing Press - Design

Sensor top dead center

Drive roller

Idle roller

Dancer

Reserve paper feed

F

G

H

B

C

I

J

Active paper feed

Flying paste

Paper cutter

Drive roller

Source: http://offsetpressman.blogspot.com/2011/03/how-flying-paster-works.html

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
This design demonstrates DC motors driving a feed roller and a drive roller. The PID-based motor controllers minimize the error between the paper velocity produced by the roller and the target profile velocity produced by the Target Profile actor. The tracking error input allows one such roller to track the other to remove small differences in paper velocity.

The target profile is either a profile from 0 to maxPaperVelocity starting at time 0 and reaching the maximum value at time Interval seconds. The profile and its derivative are continuous.

SENSOR, ACTUATOR and NETWORK ACTORS STILL NEED TO BE ADDED

Plant model

+ Distributed Controllers

DE Director
- maxPaperVelocity: 35.0
- startupInterval: 120.0
- systemSamplingInterval: 0.40
- systemStart: 0.0
- coreRadius: 0.07
- fullRollRadius: 0.7
- paperThickness: 0.000075
Demo – Contact Controller

This design demonstrates DC motors driving a feed roller and a drive roller. The PID-based motor controllers minimize the error between the paper velocity produced by the roller and the target profile velocity produced by the Target Profile actor. The tracking error input allows one such roller to track the other to remove small differences in paper velocity.

The target profile is either a profile from 0 to maxPaperVelocity starting at time 0 and reaching the maximum value at time INTERVAL seconds. The profile and its derivative are continuous.

SENSOR, ACTUATOR and NETWORK ACTORS STILL NEED TO BE ADDED

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
This design consists of a model of a controller (T-REX) and a plant mode (the rest of the design). The plant models the dynamics of a DC motors driving a feed roller and a controller to cause the resulting paper velocity to track the input profile. All of this is a subset of the larger printing press model.

The T-REX control takes as input sensors detecting top dead center of the roll, the present of the tape, and an arming signal. It outputs actuation signals: Contact to the device forcing the primary feed paper into the reserve paper, Cut to the device to cut the primary feed paper.
Demo – Printing Press

Model time delays

Sensors

Computations

Actuators
Renesas vs. XMOS: I/O timing

Contact (red), Top Dead Center (green), Cut (blue) and Arm (black)

Simulation

Renesas

XMOS

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
Renesas vs. XMOS: I/O timing

Simulation

Renesas

XMOS

Patricia Derler, MBDEFP Workshop, 10/13/2011, Taipei
Renesas vs. XMOS: Busy vs. Idle Time
Renesas vs. XMOS: Busy vs. Idle Time
Renesas vs. XMOS: Busy vs. Idle Time
Summary

- Ptides: deterministic, event-triggered model
  - Explicit model of timing
- Generate code on different platforms
  - with deterministic I/O behavior
- Unchanged I/O behavior in
  - Distributed execution
  - Parallel execution

... and if you want to try out the models, download the latest Ptolemy version and open the models in ptolemy/domains/ptides/demo/TREX and ptolemy/domains/ptides/demo/PrintingPress

Future Work

- Better strategies for parallel execution
- Scheduability analysis
- Optimization of generated code
- Driver implementations for Sensors and Actuators
- Fixed Point vs. Floating Point