Real-Time Prototyping in Ptolemy

Ptolemy Review
University of California, Berkeley
March 10, 1995

José Luis Pino and Edward A. Lee

pino@eecs.berkeley.edu
http://www.eecs.berkeley.edu/~pino

Objectives

Provide a framework to:
• Specify systems using natural models of computation
• Use hardware within a simulation
• Use the user’s computing environment
• Construct heterogeneous multiprocessor real-time prototypes
★ Shorten the design cycle

System Simulation

• Interpreted — blocks compiled into Ptolemy system
• Multiple models of computation — process networks, communicating processes, discrete event, RTL

Code Synthesis

• Compile-time scheduling
• SDF & BDF models supported with extensions that allow for nondeterminate communication
• Object-oriented target specification

CGC-S56X
Sparc
C & Tcl/Tk
M56K
Assembly
**Communication Actors**

- **Send/Receive**
  Multiprocessor self-timed SDF graphs

![Diagram of Send/Receive](Image)

- **Peek/Poke — Asynchronous & nondeterminate**
  Multiple independent SDF graphs

![Diagram of Peek/Poke](Image)

**Simulation Interface Construction**

- Uses send/receive communication actors
- Incremental compilation
- Simulation block is constructed

**Migration to Hardware**

- Top level — runs using a Ptolemy simulation domain (SDF)
- Subsystem compiled, downloaded and run on a S-56X DSP board installed in a host workstation

**Stand-alone Prototype Synthesis**

- Heterogenous multiprocessor support
- Hierarchical scheduling
- **Peek/Poke** — Extend SDF and BDF by allowing for nondeterminate communication
- Example of useful nondeterminism — real-time prototype user interfaces

![Diagram of Stand-alone Prototype Synthesis](Image)
**Peek/Poke Properties**

- Update rate is **explicit**, **implicit** or **event driven** (change of value)
- Single Sample
- Sliding Window
- Block aligned

**FM Synthesis Specification**

- DSP — Real-time engine
- Host — Spectrum calculated off-line
- DSP — Real-time engine

**FM Synthesis: GUI**

- Control panel
- Keyboard
- Time signal
- Spectrum

**Conclusions**

- Describe system with simulation domains
- Migrate subsystems to prototype hardware, generating a composite block for simulation which can be added to block library
- Generate a real-time stand-alone system using nondeterminate **peek/poke** communication actors as necessary