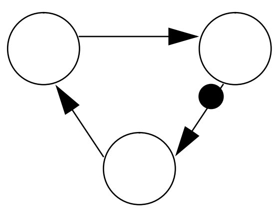
Overview of the Ptolemy Project

Brian L. Evans and H. John Reekie

Dept. of Electrical Engineering and Computer Sciences
University of California, Berkeley, CA 94720-1770

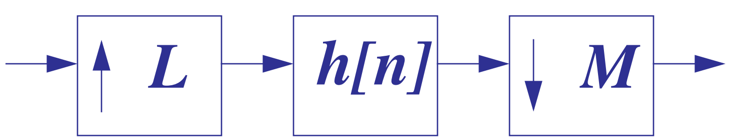
{ble,johnr}@eecs.berkeley.edu
http://ptolemy.eecs.berkeley.edu/

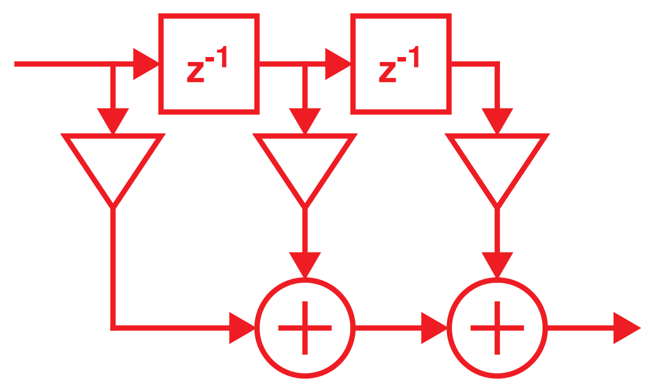


... 010010 ...

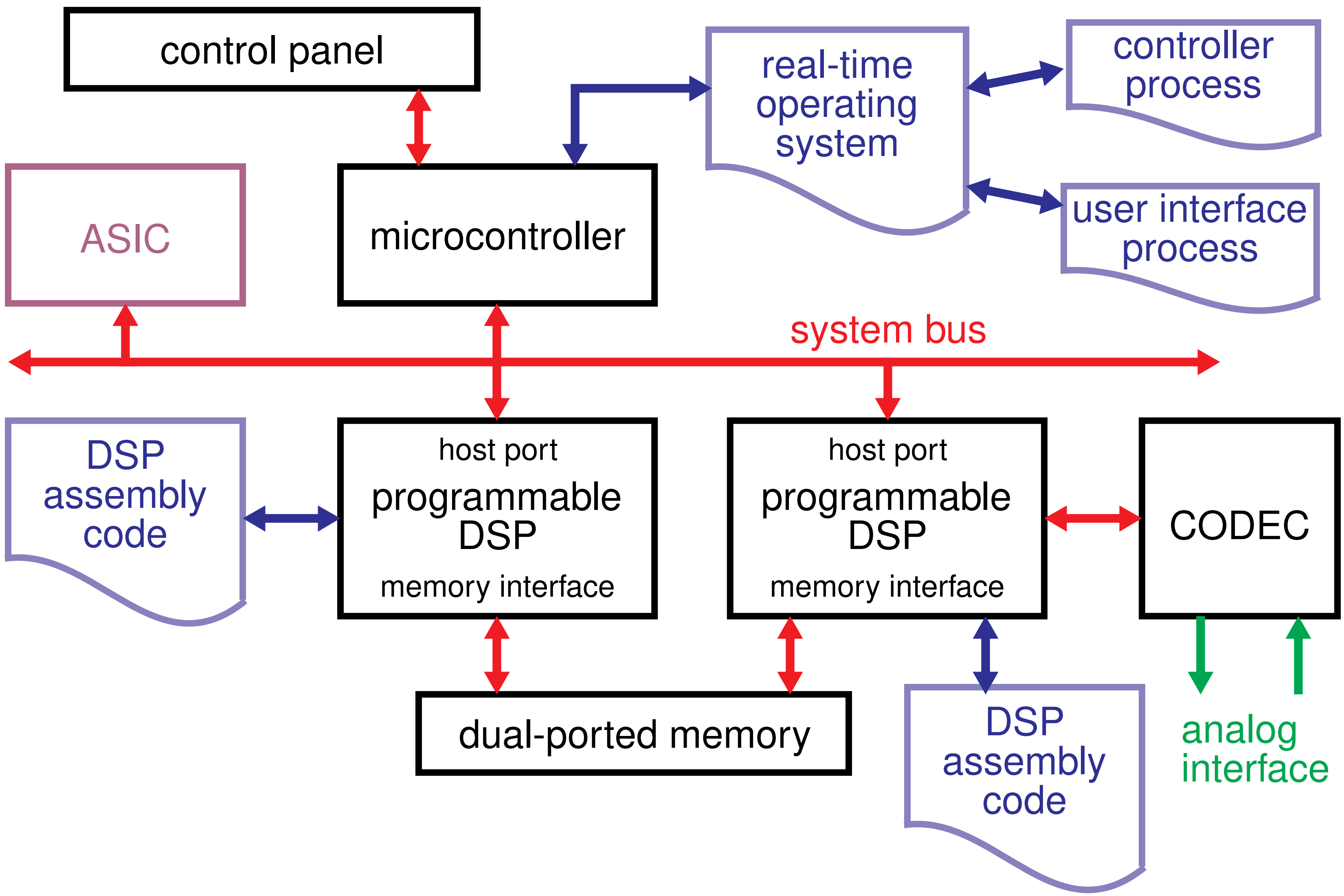
$$y[n] = h[n] * a^n u[n]$$

Project directed by Prof. Edward A. Lee and co-directed by Prof. David G. Messerschmitt

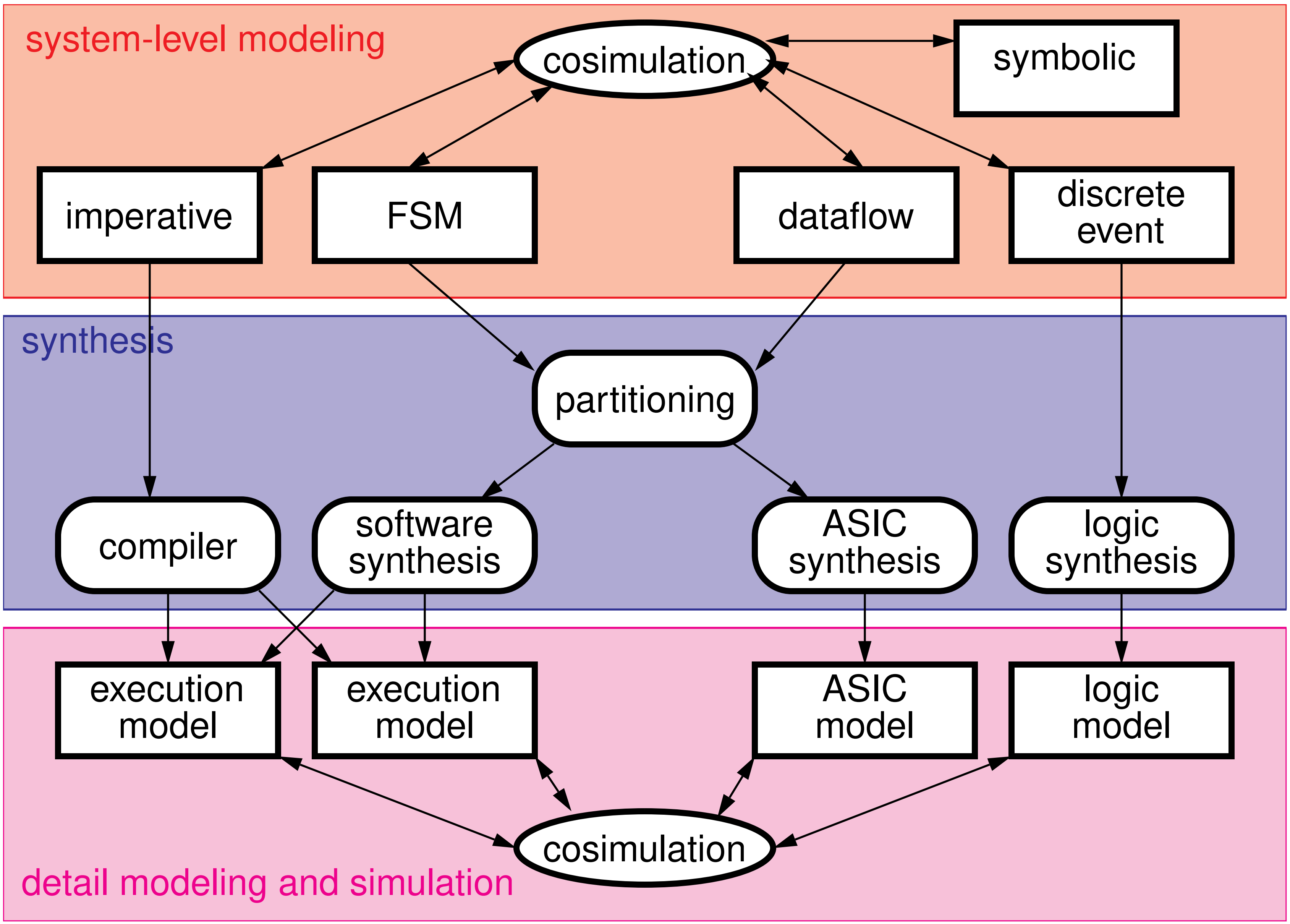




A Typical Embedded Signal Processing System



Heterogeneity in System-Level Design



Ptolemy Project

Design Methodologies for Heterogeneous Systems

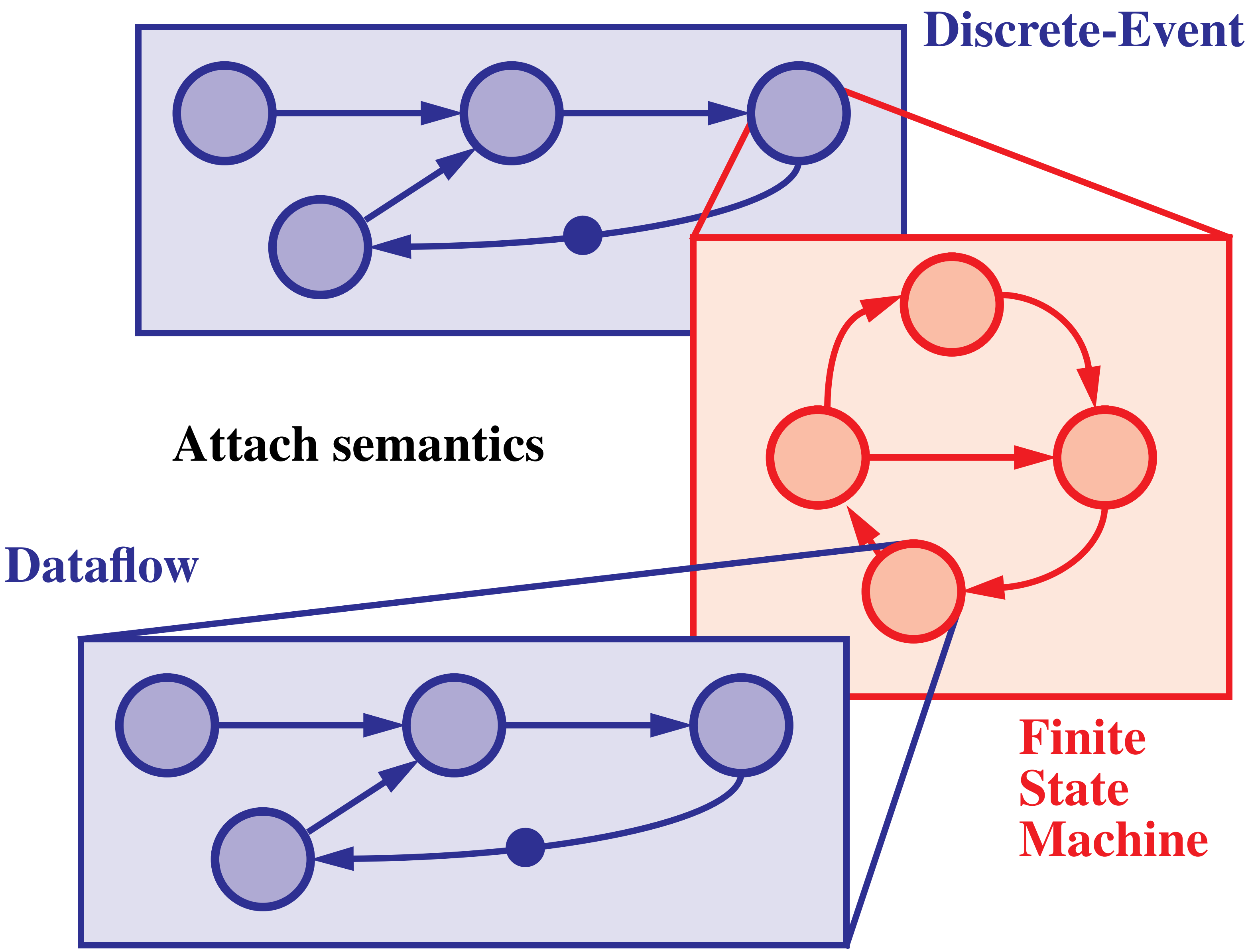
- Formal models of computation
- Hierarchical compositions of models form complex systems
- Synthesis and partitioning algorithms
- Laboratory to test design methodology is the Ptolemy software environment

Claudius Ptolemaeus

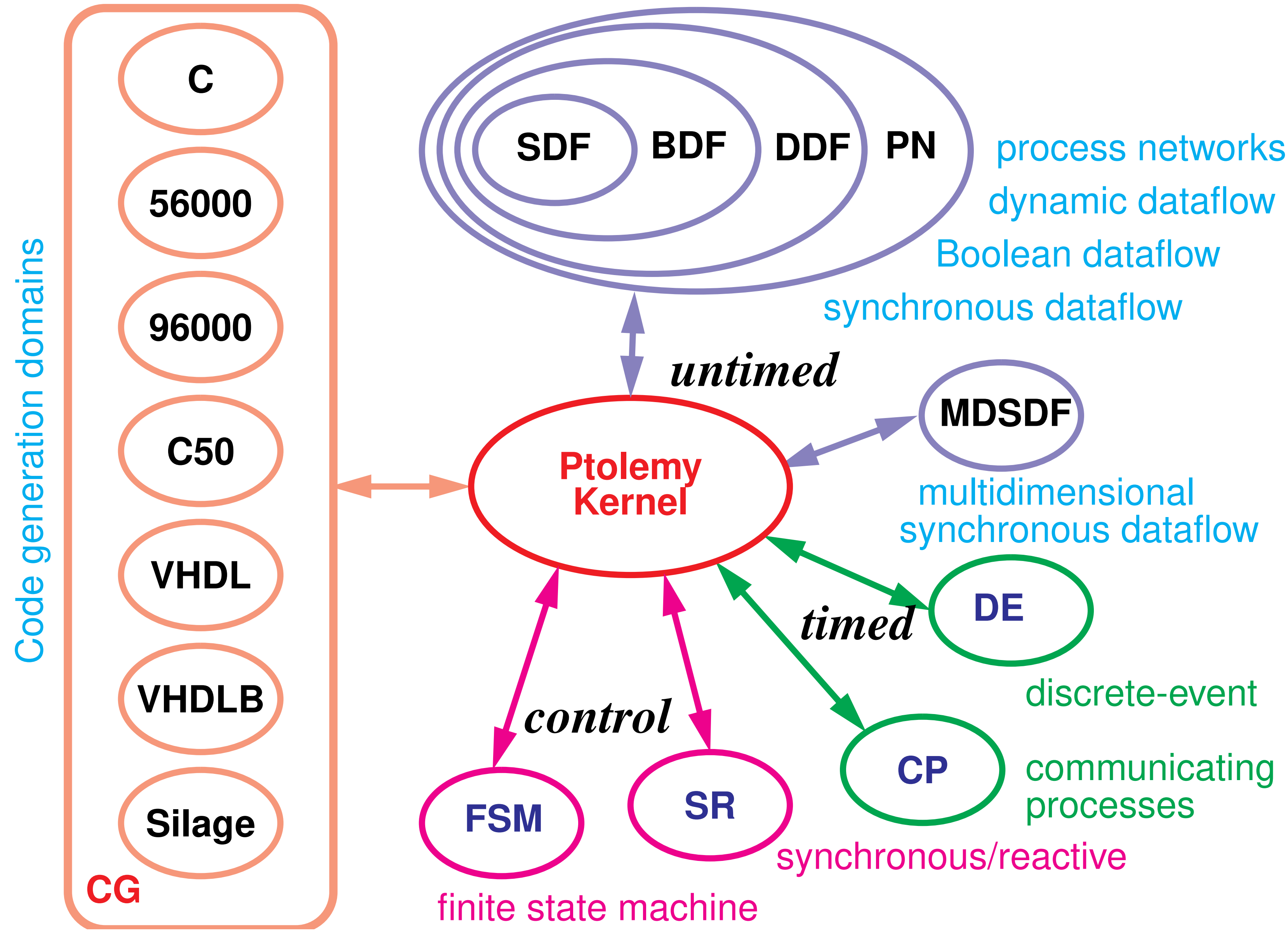
Personnel

- Directors: Profs. Edward Lee and David Messerschmitt
- Staff: 4 post-doctoral, 1 software manager, 2 administrative
- Students: 12 graduate and 3 undergraduate

Hierarchical Graphs As Underlying Abstract Syntax



Computational Models (Domains) in Ptolemy



Impact on Industrial CAD Tools

Source	Year	Product
Synchronous Dataflow model plus multirate schedulers	*1992*	Signal Processing WorkSystem from Cadence
Heterogeneous simulation support by Ptolemy kernel	*1995*	CONVERGENCE environment from Cadence
Heterogeneous simulation support by Ptolemy kernel	*1995*	Ptolemy HSIM from Berkeley Design Technology, Inc.
Synchronous Dataflow and Discrete-Event domains	*1995*	Sanders ENTIRE framework (multiprocessor architecture mapping and FPGA synthesis)
Ptolemy kernel plus dataflow domains	*1996*	Thomson-CSF radar simulator

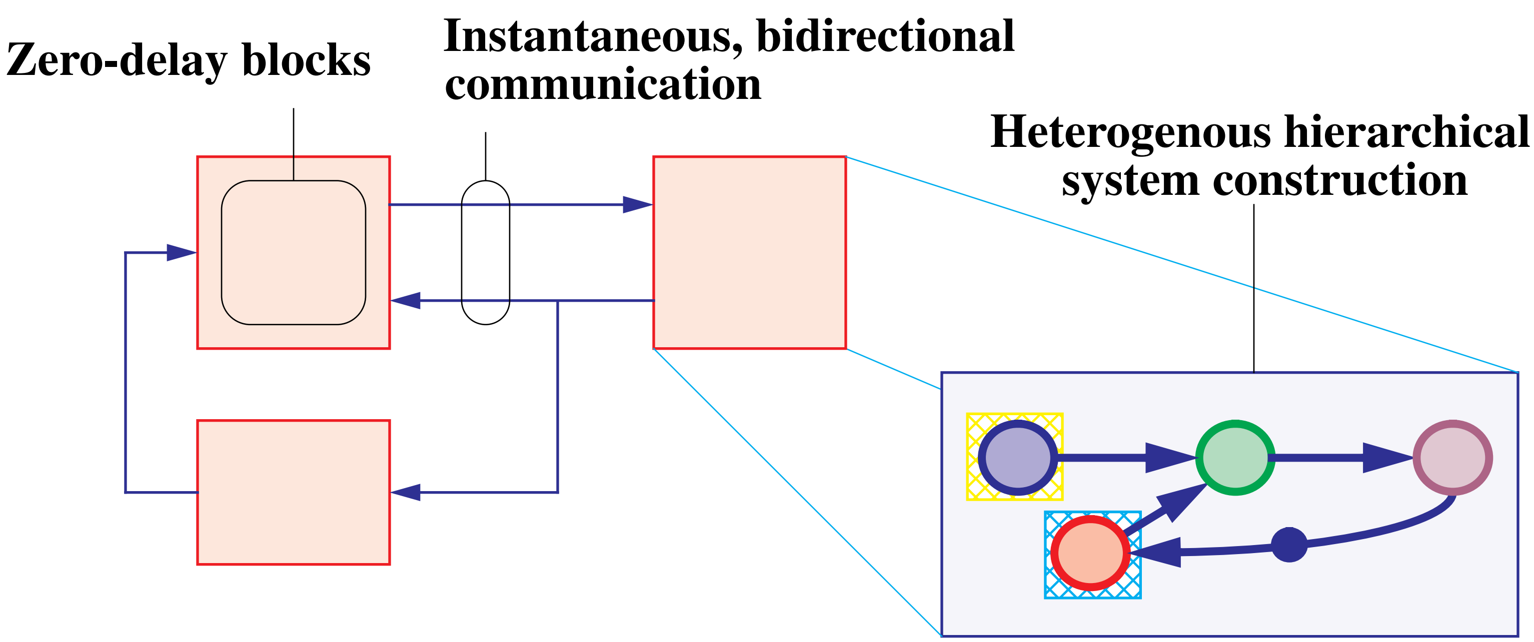
http://ptolemy.eecs.berkeley.edu/third-party.html

Our Current Research in System-Level Design

	Topic	Example
Specification	System optimization	System rearrangement
	Scalable systems	Higher-order functions
	Visual languages	Tycho
	Formal semantics	Tagged signal model
Simulation	Dataflow models of computation	Multidimensional dataflow
	Control models of computation	Synchronous/reactive
	Cosimulation	Mixed signal simulation
	Native signal processing	UltraSparc visual instruction set
Synthesis	Partitioning	Hardware/software codesign
	Uniprocessor scheduling	Program/data code minimization
	Multiprocessor scheduling	Hierarchical scheduling
	Distributed systems	Network of workstations

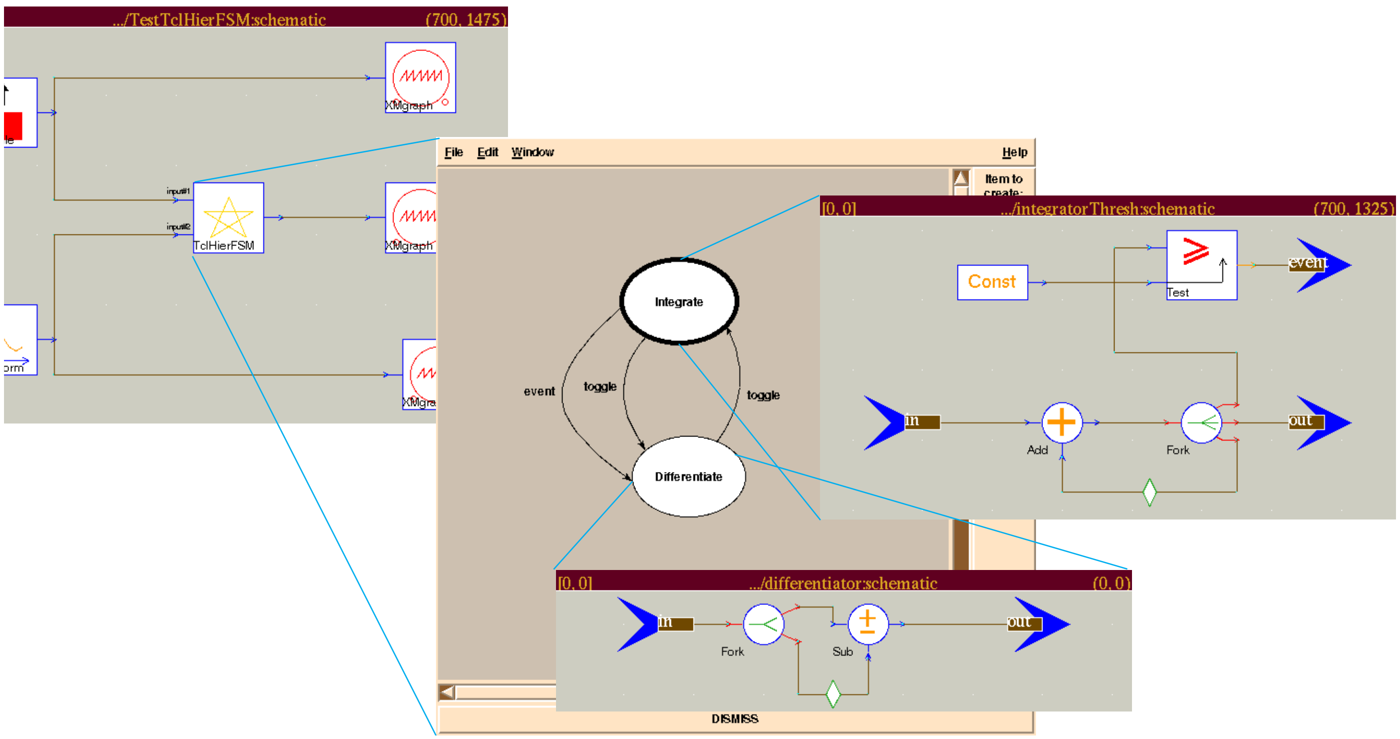
Synchronous/Reactive Model of Computation

- Synchronous means that computations occur "instantaneously" at integer clock ticks (Lustre, Signal)
- Reactive means that the model responds to the environment at the speed of the environment (Esterel, StateCharts)
- Modules are monotonic functions operating on complete partial orders
- Execution proceeds by iterating towards the fixed-point — compile-time analysis finds an execution order guaranteed to produce the least fixed-point
- Schedules determined at compile-time in polynomial time



Mixing FSMs and Dataflow Models of Computation

- Each Finite State Machine (FSM) has a single thread of control
- Arranging FSMs in a nested tree specifies hierarchical FSMs
- Arbitrary mixing of FSMs and dataflow models of computation captures captures the 21 variants of Statecharts
- Formal analysis (verification) possible due to finite state



Vision for Common Operating Environments

Requirements to avoid

- Avoid one monolithic standard
- Avoid standardizing on one general purpose language
- Avoid one specification format

Requirements to include

- Support of domain-specific models of computation/tools
- Support imperative and declarative styles of programming
- Support multiple specification formats, such as directed acyclic graphs, textual languages, algebraic descriptions
- Support back annotation
- Support general frameworks for tools to interface with
- Support cosimulation of arbitrary levels of abstraction