

Application of Formal Methods and their Analysis in Embedded Systems

By:

Chaitanya Settaluri

Devendra Kalia



What is an embedded system?

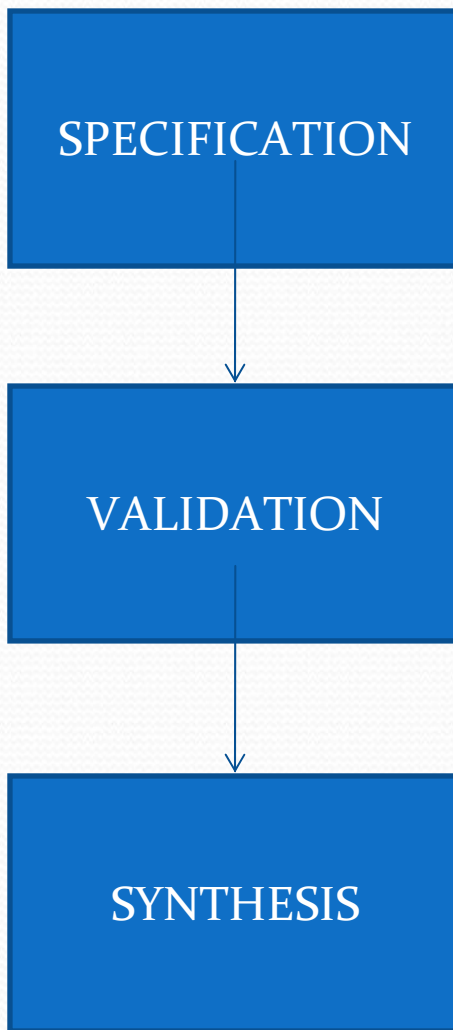
- **An embedded system**
 - Uses a controller to perform some function
 - Is not perceived as a computer
- **Software is used for features and flexibility**
- **Hardware is used for performance**
- **Typical characteristics**
 - It performs a single function
 - It is part of a larger (controlled) system
 - Cost and Reliability are often the most significant aspects



Flow of Presentation

- Embedded systems
- Specification
- Models of computation
- Validation
- Simulation
- Formal verification
- Synthesis
- Partitioning
- Hardware and software synthesis
- Conclusion

Refinement Phases



Need for formal specification?

- In the development of embedded reactive systems the specification of the requirements is most critical issue.
- Embedded systems – Demand a high degree of reliability when running in risk critical applications
- Primary purpose- To provide clear and unambiguous description of system

COMPONENTS OF FORMAL MODELS OF DESIGN

- **FUNCTIONAL SPECIFICATION** – relations involving inputs, outputs and state information
- **PROPERTIES**- relations over inputs, outputs and states that can be checked against functional specification
- **PERFORMANCE INDICES**- these evaluate quality of design in terms of cost, reliability, speed, size
- **CONSTRAINTS**

Models of Computation

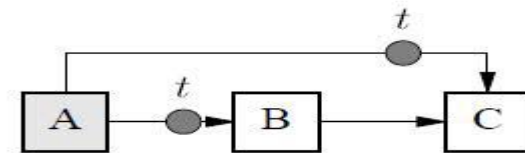
- Definition: A mathematical description that has syntax and rules for computation of the behavior
- Features:
 - a. Expressiveness
 - b. Generality
 - c. Simplicity
 - d. Synthesizability
 - e. Verifiability

DISCRETE EVENT MODEL

- Events carry a totally ordered time stamp which is discrete.
- Processes are executed when they receive input events and produce output events

Disadvantages:

- Distributed Execution is difficult
- No specified order of events for simultaneous events



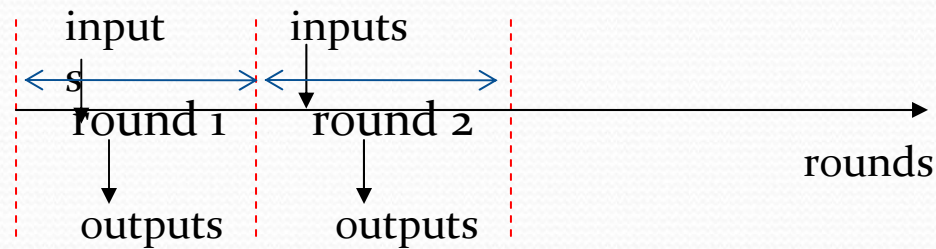
Simultaneous Events

Communicating FSM

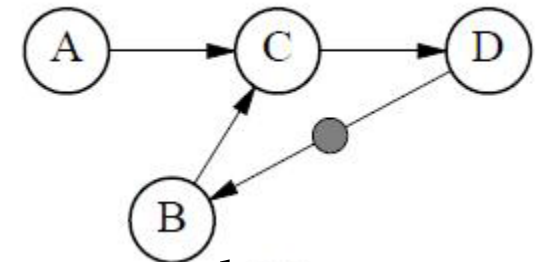
- FSM is represented by quintuple $(\Sigma, S, s_0, \delta, F)$
- Synchronous FSM: Communication is by shared variables
- Asynchronous FSM: Communication is by lossless channel
- Suffer from State Explosion Problem

Synchronous Model

- Synchronous events
 - Signals have events with identical tags
 - Order of processing of events-determined by Data Precedence
- ...



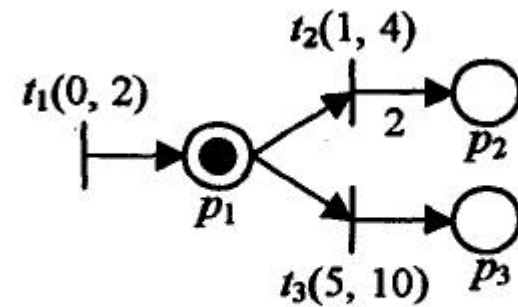
Data Flow Networks



- Dataflow actor
 - When it fires, it maps input tokens into output tokens
- Firing
 - Consumes input tokens and produces output tokens
 - A sequence of such firings is a particular type of Kahn process that we call a dataflow process;
 - A network of such processes is called a dataflow process network.
- Firing rules
 - Specify when an Actor can fire
- Deterministic behavior of the model because of the blocking constraint

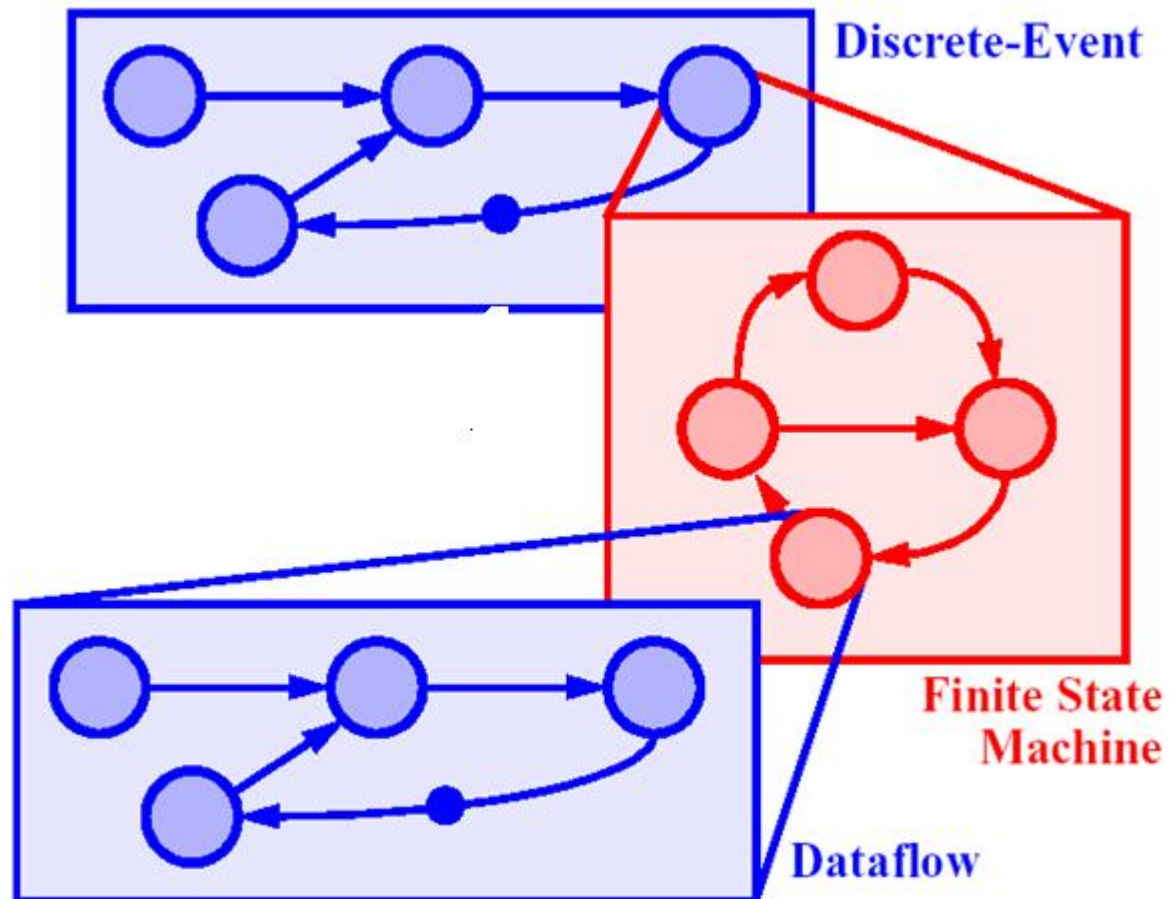
Time Free-Choice Petri Nets

- TFCPN is represented as a 5-tuple (P,T,F,M,τ)
- Pre-specified number of tokens in the input places fires a transition
- Quasi-Static Data Scheduling is used to address Bounded-Memory Execution
- Dynamic Real Time Scheduling is used to address Real Time Constraint



A TFCPN
representation

Heterogeneous Models



Observations

- Any Embedded System described using a particular model satisfies the properties inherent in the model
- Some properties like determinism can be verified semantically by checking for all the inputs

Validation

- Validation refers to the process of determining that a design is correct
- The main tool for validation has always been simulation
- But formal verification is gaining importance, especially for safety-critical embedded systems

System validation

- Safety-critical real-time systems *must* be validated
 - Explicit exhaustive simulation is infeasible
 - Formal verification can achieve the same level of safeness
- How to use verification and simulation together ?
 - Simulation can be used initially for
 - Quick functional debugging
 - Eliminating obvious cases
 - Then formal verification is done for exhaustive checking
 - Finally simulation is used again as *user interface* to provide the designer with *error traces*

Simulation

- Simulation is the operation of a real-world process or system over time
- Simulation involves the generation of an artificial history of the system
- Inferences are drawn and real system is represented

Simulation

- Simulating embedded system is challenging because they are heterogeneous
 - Both software and hardware components must be simulated at the same time (*the co-simulation problem*)
 - To test software as fast as possible are used machine that may be faster the final embedded CPU, and is very different from it
 - Necessary to keep the hardware and software simulation synchronized, so that they interact just as they will in the target system
- A solution is to use a general-purpose software simulator to simulate a model of target CPU

Co-simulation methods

- A unified approach, where the entire system is translated into a form suitable for a single simulator, is conceptually simple, but computationally inefficient.
- Making better use of computational resources often means distributing the simulation
- But synchronization of the processes becomes a challenge.

Formal verification

- Verification can be divided into two types :
 - **Specification Verification:** checking an abstract property of a high-level model
 - Example: checking whether a protocol modeled as a network of communicating FSMs can ever deadlock
 - **Implementation Verification:** checking if a relatively low-level model correctly implements a higher-level model or satisfies some implementation-dependent property

Embedded system verification

- Theorem proving methods
- Finite automata methods

Finite automata methods can be of two types :

- Containment method
- Model checking

- Infinite automata methods

Synthesis

- During synthesis a less abstract specification is created or implemented
- For embedded systems, synthesis is divided into three stages
 - Architecture definition, structure for implementation is defined
 - Partitioning, Mapping from specification to architecture is done
 - Hardware and software synthesis, Implementation of hardware and software components is carried out

Partitioning

- Partitioning is a problem in embedded systems which arises due to heterogeneous hardware and software components
- In partitioning the specification parts are mapped to architecture components
- No optimal partitioning method has been found yet



Hardware and software synthesis

- It is important to implement hardware and software synthesis in order to estimate cost function
- The input specification is converted to an implementation in form of hardware and software
- The objective is to realize the specification with the minimum cost

Acknowledgements

- An introduction to **Embedded Systems**
Michele Arcuri *Software Engineering 2 A.A. 2001-2002*
- Design of Embedded Systems: Formal Models, Validation, and Synthesis
By : S. Edwards, L. Lavagno, E. A. Lee, and A. Sangiovanni-Vincentelli