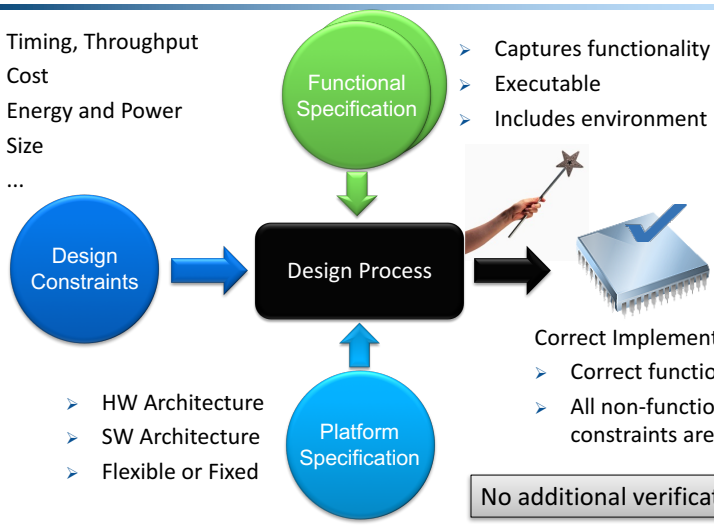
Formal Design of Mixed-Criticality Systems

Ingo Sander
Department of Electronics
School of Information and Communication Technology
KTH Royal Institute of Technology
Stockholm, Sweden
ingo@kth.se

Correct-by-Construction Design The Dream...

- Timing, Throughput
- Cost
- Energy and Power
- Size
- ...



- Captures functionality
- Executable
- Includes environment

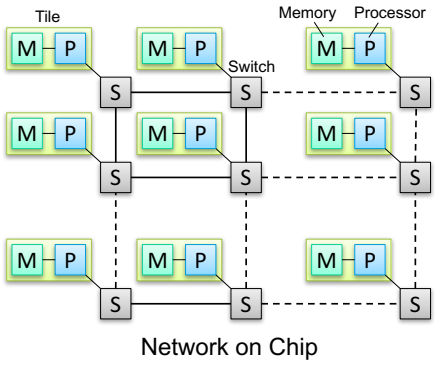
Correct Implementation
- Correct functionality
- All non-functional constraints are fulfilled

- HW Architecture
- SW Architecture
- Flexible or Fixed

No additional verification needed!

Design Challenges: The Many-Core Era

- Moore's law leads to
  - increasingly parallel, powerful and complex architectures
  - increasingly advanced and demanding applications
- Real-time requirements are difficult to verify



We have already today problems with complexity, so how will we design tomorrow's "sea-of-cores" real-time systems?

Learn from Hardware Design

- Hardware design is predictable
  - Based on simple communication mechanism (clock)
  - Formal models are available (Boolean algebra and synchronous model)
  - Predictable architecture (gate delays)

⇒ Very efficient tools!

  ⇒ Timing predictable in range of nano- or pico seconds

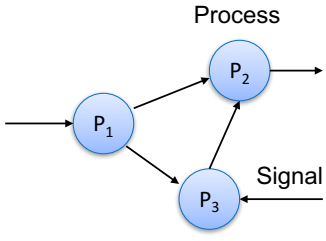
Synchronous languages (Esterel, Lustre) have been very successful for safety-critical applications!

Embedded Real-Time Software What is needed?

- Formal analyzable models of the application
- Platforms that can provide service guarantees
- Analysis methods enabling design of tools
- (Industrial) entry language enabling
  - Simulation of the application model
  - Automatic extraction of analyzable models

Models of Computation (MoC)

- MoC specifies semantics of computation and communication
- Abstracts from design language
- Enables application of analysis methods
  - Design space exploration
  - Performance analysis
  - Verification
  - Synthesis
- Active research area
  - Development of new techniques and tools

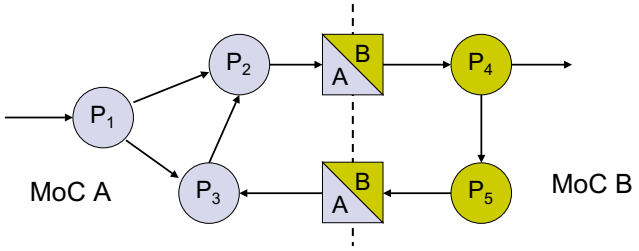


ForSyDe Methodology

- ForSyDe (Formal System Design) targets design of heterogeneous safety-critical embedded systems
- Formal basis
  - Models of computation theory
  - Functional programming paradigm
- Idea
  - Use higher-order functions to construct processes for different models of computation
  - Provide designers with libraries ensuring that system models are consistent with model of computation
  - ForSyDe is language independent
    - Current Implementations: Haskell, SystemC

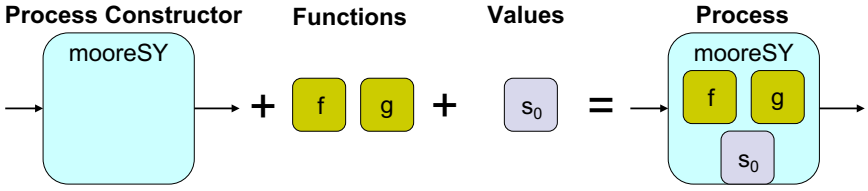
ForSyDe (Formal System Design) System Model

- A system is modeled as hierarchical concurrent process model
- Processes of different models of computation (MoC) communicate via MoC interfaces
- ForSyDe libraries support the designer in the development of an executable formal model



Designing in ForSyDe
Processes

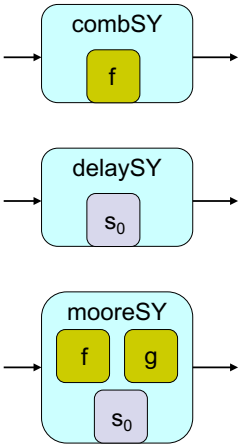
- A process is always designed by means of a process constructor
- The process constructor defines the communication interface of the process
- The process constructor takes side-effect free *functions* and *values* as arguments and returns a deterministic process



ForSyDe processes and system models are deterministic!

Designing in ForSyDe
Process Constructor Benefits

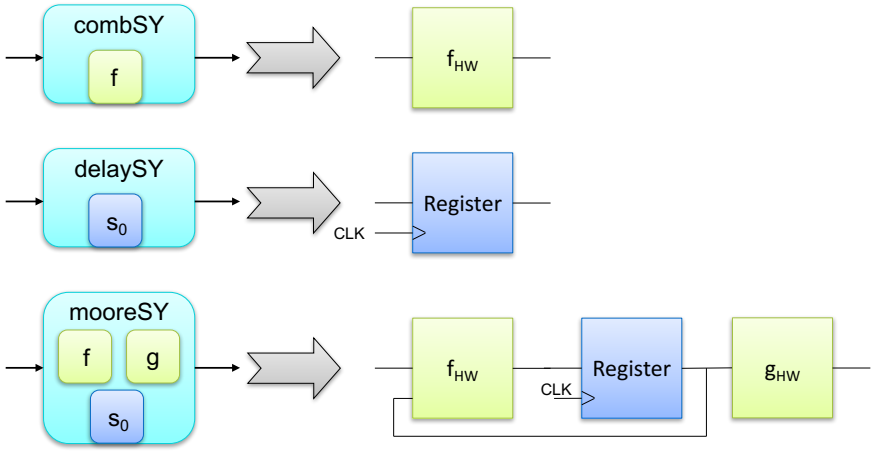
- The concept of process constructor
  - separates communication and computation
    - communication and interface: process constructor
    - computation: function
  - forces the designer to develop a structured formal model that allows for formal analysis
  - allows to define "mappings" to implementation languages



Three basic types of process constructors

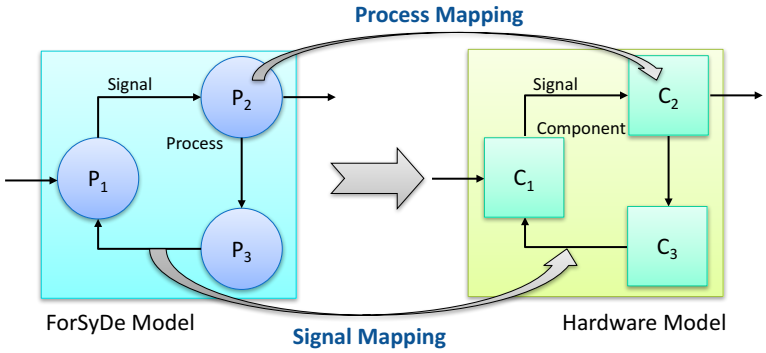
Implementation Mapping
Example: Hardware Synthesis

- Synthesis of Process Constructors



Implementation Mapping
Example: Hardware Synthesis

- Synthesis of Process Network

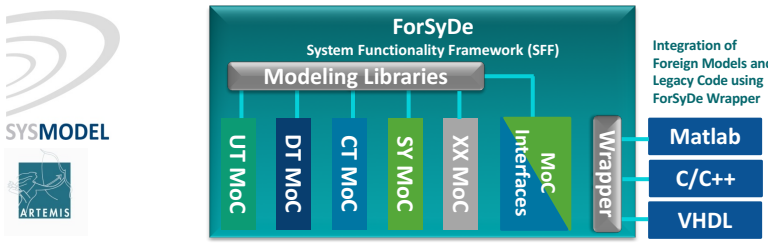


General ForSyDe Synthesis Principles

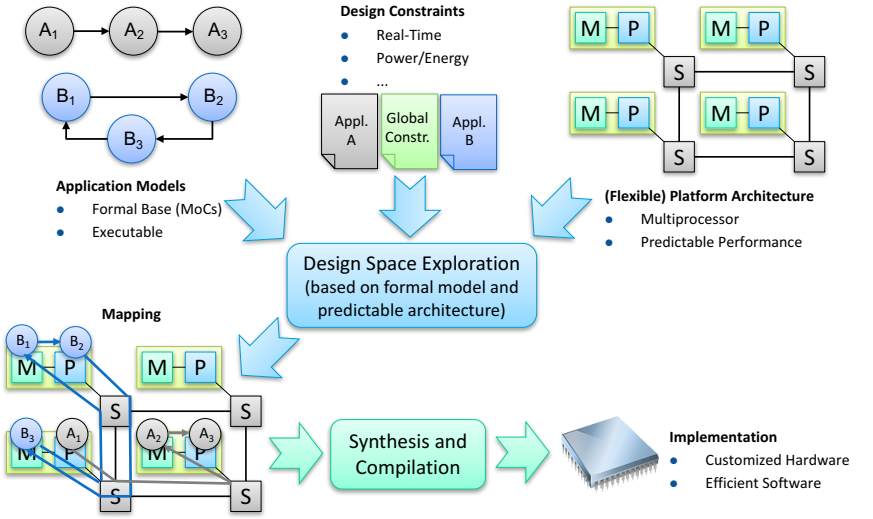
- To synthesize a ForSyDe model into a target implementation, the following synthesis mechanisms have to be developed
  1. Synthesis of concurrent process networks
  2. Synthesis of processes created by process constructors
     - Synthesis of process constructors
     - Synthesis of process constructor arguments
  3. Synthesis of basic process and model interfaces

SystemC ForSyDe

- SystemC libraries force the designer to develop a structured model enabling *formal analysis*
- SystemC-wrappers are used to
  - integrate existing models written in other languages
  - co-simulate low-level models running on executable platforms with high-level models



Design Flow: Mixed-Criticality Systems



Designer Perspective: ForSyDe Design Flow

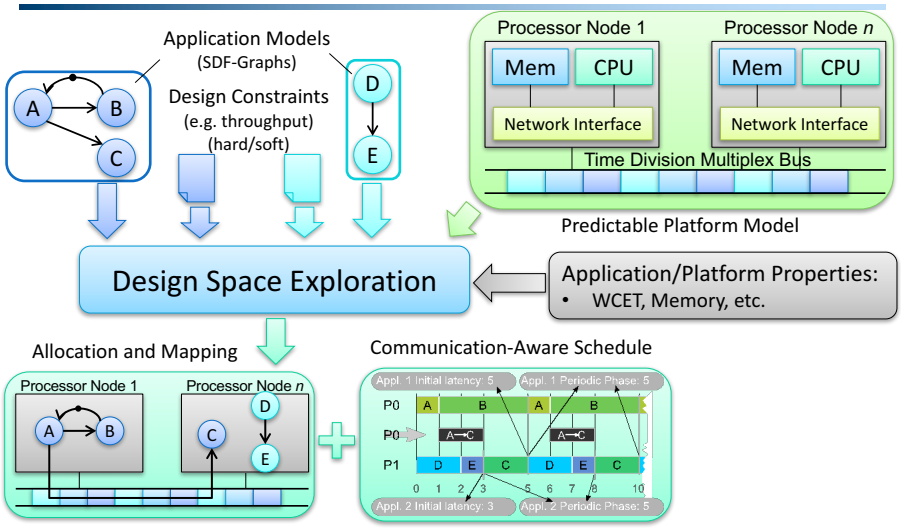
Designer models Executable ForSyDe Model

Extraction of Analyzable Model (XML+C)

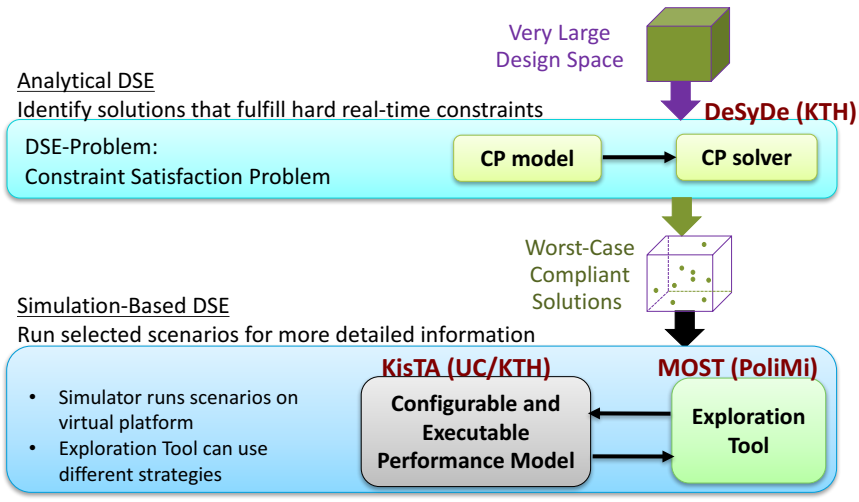
Design Space Exploration ➔ Efficient Mapping

Code Generation for Individual Processors

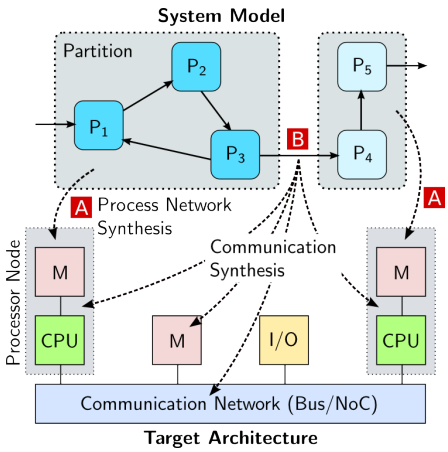
Design Space Exploration DeSyDe Tool



Joint Analytical and Simulation-based DSE



Synthesis to Target Platform



Process Network Synthesis

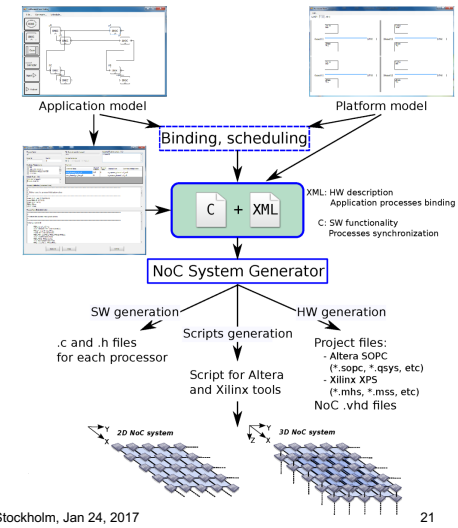
1. Implementation of static schedule
2. Generation of internal FIFO buffers
3. Code generation for each process, including function calls for message passing communication

Prerequisites

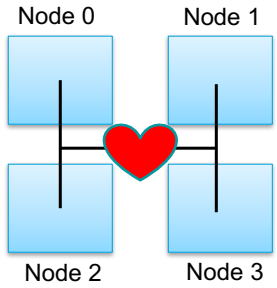
- Platform provides communication primitives for message passing and service guarantees

Network-on-Chip System Generator

- Application Model
  - Concurrent process network (Synchronous MoC)
  - Process functionality in C
- Platform Model
  - 1D, 2D- and 3D-matrix of tiles
  - Tiles: Single bus, support for multiple CPUs, memories, peripherals and IP-cores
  - Heartbeat to support synchronous MoC



The Heartbeat Methodology for Real-Time Systems Design



The Heartbeat provides the pace for the PE's.

The node generating the pace is referred to as the PaceMaker.

The pace dictates when ***Synchronous*** Processes exchange/see data.

ForSyDe Current Work

- Development of automated design flow for mixed-criticality systems
  - ForSyDe System Modelling Environment
  - Design space exploration method based on constraint programming (DeSyDe)
  - Software synthesis of ForSyDe system models to Heartbeat NoC
  - Integration of low-power techniques
- ForSyDe tools are publicly available on github: https://github.com/forsyde

EMC² Contrex SAFEPOWER