SDF3: dataflow analysis and mapping

Sander Stuijk, Marc Geilen, Twan Basten, Bart Theelen
Embedded streaming systems

Application trends
- Dynamism
- Concurrency

Predictability
The timing behavior of an application can be guaranteed independent of other applications running in the system

[Thanks to Martijn Koedam]
Model-based design

Modeling

Analysis

Implementation
- Model abstract from dynamic behavior
- Many efficient design-time analysis algorithms available
- Low implementation overhead
- SDF-based design approach may lead to resource over-allocation
Scenario-aware dataflow

- Dynamic behavior captured in scenarios
- Applications have relatively static behavior inside a scenario
- Trade-off between number of scenarios, run-time analysis techniques, and implementation efficiency

\[ x = \{0, 30, 40, 50, 60, 70, 80, 99\} \]
Similar to SDF, an *iteration* is a set of actor firings that have no net effect on the token distribution.

Different variants of SADF can model different scenario switching behavior.
Scenario-aware dataflow – scenario switching

- FSM-based SADF
  - Scenario executed for complete iteration
  - Each scenario corresponds to an SDF graph
  - FSM specifies possible scenario sequences
Analyzing SADF graphs

- Analysis techniques
  - Throughput
  - Latency
  - Buffer requirements
- Techniques based on (max,+)-algebra

- Assumption
  - Relevant implementation aspects must be modeled in the graph

- Example: scenario aware dataflow graph with a static structure
  - Execution times vary with scenarios a and b
SDF$^3$: SDF For Free

- Key features
  - Open-source GPL licensed software
  - Separation of analysis, transformation, and implementation techniques
  - Additional MPSoC platforms can be added with minimal effort
Summary

- SADF Model-of-Computation
  - Scenarios capture dynamic (application) behavior
  - Provides many analysis techniques
  - Provides implementation trajectory

- Dataflow graph model captures
  - Application behavior
  - Timing impact of platform resources

- Use of single MoC enables model-based design of predictable systems

- Analysis and implementation techniques implemented in SDF$^3$ tool kit

www.es.ele.tue.nl/sdf3
Hands-on session

Sander Stuijk

Department of Electrical Engineering
Electronic Systems
Hands-on session

- Archive with all files be downloaded from
  www.es.ele.tue.nl/~sander/tutorials/sdf3-2012/sdf3-tutorial.zip

- USB sticks with files are also available

- Instructions
  - Unpack archive to location of your own choice
  - Open the file index.html in a web browser
  - Select option ‘hands-on exercises’ from top menu bar