The SystemJ Approach to System-level Design

Flavius Gruian¹, Partha Roop², Zoran Salcic², Ivan Radojevic²

1. flavius.gruiian@cs.lth.se
Dept of Computer Science
Lund Institute of Technology
Sweden

2. {p.roop,z.salcic,i.radojevic}@auckland.ac.nz
Dept. of Electrical and Computer Engineering
The University of Auckland
New Zealand

SystemJ, the language designed for specification, modeling and synthesis of GALS systems

- **Java**
  - object orientation
  - basic data and control processing
  - platform support: GCC compilers, libraries
  - desktop and embedded use

- **TReK**
  - a True Reactive Kernel
  - Java 1.5 Library
  - offers support for desktop execution of SystemJ specifications
  - sixteen classes implementing reactions, signals, channels, clock domains, and scheduling
  - extends standard Java threads and employs two queues to achieve the synch-async behaviour at run-time
  - uses try-catch to handle aborts and traps
  - signals and signal operations implemented using bit arrays
  - employs generics to implement valued signals and channels

- **Asynchronous Constructs** (Esterel-like)
  - composition
  - channel operations
  - pure/valued signals
  - preemptions

- **Synchronous Constructs**
  - composition
  - channel operations
  - pure/valued signals
  - preemptions

**Possible uses of SystemJ**

- Java based design flows

**Ongoing & Planned Work**

- reactive Java optimized processor (a JOP-ReMIC mix)
- multi-processor architectures (based on HiDRA and Emperor)
- custom JVM for efficient scheduling
- debug and visualisation tools in Eclipse
- formal semantics

**Formal Verification**

- no formal semantics YET
- can be translated to extended CRSM

**The SystemJ Pre-processor**

- translates SystemJ to Java plus TReK calls
- detects syntactic errors, incorrect uses of channels and signals
- introduces signal resolves to help the run-time scheduler
- based on ReRAGs with JastAdd, a Java 1.4 frontend and generates Java 1.5 code

**Examples of SystemJ constructs translated to Java/TReK**

**The equivalent ECRSM of the Protocol Stack Example**