Mapping Multiple Independent Synchronous Dataflow Graphs

Presented at the Twenty-Eighth Annual Asilomar Conference on Signals, Systems, and Computers - October 1994

José Luis Pino, Thomas M. Parks and Edward A. Lee
Goal

• Automated code generation of real-time applications for heterogeneous architectures

Method

• New model of computation that allows nondeterminate communication between independent dataflow graphs
Target Specification

• Hierarchical object-oriented target specification
  — Parent and child targets
  — Information hiding

• Compile-time scheduling

• Synthesize C and assembly code
Application Specification

Synchronous Data Flow (SDF)

AABAB

Multiple Independent SDF Graphs

3(A)2(B) CDE

48 kHz

44.1 kHz
Communication Actors

- **Send/Receive**
  Multiprocessor self-timed SDF graphs

- **Peek/Poke**
  Multiple independent SDF graphs (multiprocessor and/or uniprocessor systems)
Peek/Poke Properties

- Update rate is **explicit, implicit or event driven** (change of value)
- Single Sample
- Sliding Window
- Block aligned
Static Scheduling

• Must know relative rates of graphs
  — Explicitly: Specified by user
  — Implicitly: Derived from real-time actors

• Static schedule for example:
  — Concatenated: 160(3(A)2(B)) 147(CDE)
  — Interleaved: 13(3(A)2(B)) 147(3(A)2(B)CDE)
Dynamic Scheduling

- Relative firing rates may not be exactly known (i.e. driven by separate hardware clocks)
- Might need preemption (i.e. execution time of one actor may exceed the period of another real-time actor)
- **Rate-monotonic** priority assignment for the independent graphs scheduled dynamically using a real-time operating system
FM Synthesis: Specification

Chowning FM Synthesis
FM Synthesis: GUI
Acoustic Modem: GUI
Conclusions

• Extended synchronous dataflow with nondeterminate peek/poke communication actors

• Static scheduling for implicit or explicit graph rates

• Dynamic scheduling for unknown graph rates

• Ideal for interactive controls and displays
Future Work

• Hierarchical scheduling framework to mix
  — Multiprocessor schedulers (general and specialized)
  — Uniprocessor schedulers

• Efficient real-time dynamic scheduling
  — prioritized multithreaded execution
  — non-preemptive rate monotonic scheduling