Combining P2 and RDL to build Dataflow Hardware Programs

http://ramp.eecs.berkeley.edu
Greg Gibeling, Nathan Burkhart and Andrew Shultz
{gdgib, nburkhart, alshult}@berkeley.edu
5/1/2006

Outline

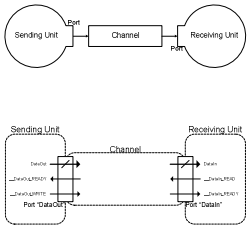
- RAMP Architecture & Target Model
- Tools & Toolflow
- RAMP Description Language
- P2
- Status & Future Work

RAMP Architecture

- Target
  - The system being emulated
    - Actually only a model of the system being emulated
    - Can be a cycle accurate model
  - Must conform to the RAMP target model
- Host
  - The system doing the emulation
  - May include multiple platforms
    - Hardware – BEE2, XUP, CaLinx2
    - Emulation – Matlab, ModelSim
    - Software – C++, Java

RAMP Target Model (1)

- Units communicate over channels
- Units
  - Implemented in a "host" language
  - Smaller than a DB Op
- Channels
  - Unidirectional
  - Point-to-point
  - FIFO semantics
  - Similar to the Exchange operator from Volcano

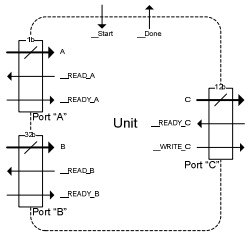


RAMP Target Model (2)

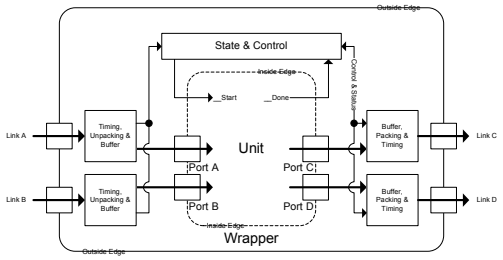
- Transaction style unit semantics
  - Read 0/1 messages from each input
  - Perform some action
  - Write 0/1 messages to each output
  - Units must be latency insensitive
- This affects the way SHIPs are coded

Target Model - Units

- Inside edge
  - Ports connect units to channels
    - FIFO signaling
    - Hardware or Software
  - Target cycle control
    - __Start
    - __Done
    - Allows for variable timing, and timing accurate simulation



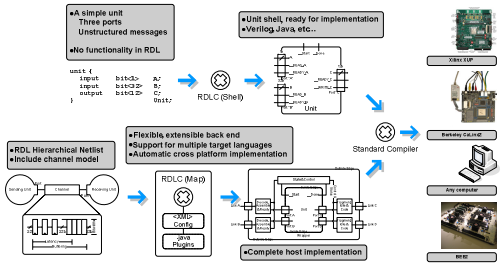
Host Model – Wrapper (2)



RDLC2 Toolflow (1)

- Development Steps
  - Unit Implementation
    - RDL unit descriptions
    - RDLC generates shell code in a specific language
    - Researcher adds implementation
  - RDL target design
    - Includes Mapping
    - RDLC generates complete implementation code
    - Includes all links, instantiates all unit shells

RDLC2 Toolflow (2)



RDLC Toolflow (3)

- Help
  - rdlc –help
  - Explains commands
  - Includes all the options
- GUI
  - rdlc –gui
  - Easy to use
  - Includes error message display

RDL (1)

- "RAMP Description Language"
- General message passing system description language
  - Hierarchical Netlisting Language
- Compiler includes back-end extensibility
  - Can invoke ModelSim/XFlow back ends
- Does not include behavioral code

RDL (2)

- Hierarchical Namespaces
  - Declarations can be external to a namespace
  - Allows for communal development
- RDL Target Constructs
  - Channels, Messages and Port types
  - Units include instances, inputs, outputs and connections
- RDL Host Constructs
  - Links and Terminal types
  - One platform per board or computer
  - Platforms include an implementation language
- RDL Mappings
  - Hierarchy allows for "compile one, run many"
  - Allows specific units and channels to be precisely mapped

RDL (3)

```
Example RDL here
```


A FLEET in RDL (1)

State of the Project

- Working hardware implementation!
  - Compiled RDL to Verilog
  - Tested on CaLinx2, XUP, Digilent S3 and ModelSim SE
- RDL & RDL Compiler
  - RDLC2 is stable
  - Working compiler, written in java
  - 125,000 lines of code
  - Definitely going to get cleaned up

Future Work (1)

- RDL & RDLC Features
  - Language Features
    - Generated code and compile time parameters
    - Support for expressions and better parameters
  - Languages, platforms, links
  - Debugging automation
- Documentation
  - Architecture, Language & Compiler Technical Report
  - Complete compiler internals documentation
  - Example and Tutorials

Future Work (2)

- Stuff