Block Oriented Network Simulator - BONeS

Marko Luoma

BONeS

- BONeS is a part of Cadence / Altagroup design tools intended to be used in top to down design paradigm.
- BONeS covers system performance analysis and some behavioural algorithm design.
- BONeS offers high level libraries which make possible to do:
  - Rapid evaluation of system ideas in early phases.
  - Through simulation of entire systems with all functionality.
BONeS Suite of Tools

Environment and User Interface
Libraries

- Physical containers for BONeS objects
  - The BONeS design database
- Control
  - Open/closed
  - Access permissions
  - Read/write
- Environment = open libraries
  - Can select only objects from open libraries
  - Can automatically set up (Options-Save Environment)
- Corresponds to UNIX directory
- System-Supplied/User
Library operations

Projects

- Another kind of container
  - Strictly a logical grouping
  - Stores references to objects
  - Shows relationships
  - Orthogonal to libraries
    - Can be several projects in a library
    - A project can reference objects from several libraries
Groups

- Another container for modules
  - Strictly logical
  - Only in BDE
  - Each object appears only once in group hierarchy
  - Groups form a tree

- Main window selection
  - By library
  - By group

Block Diagram Editor (BDE)

- Block diagram editor is the tool used to create modules and systems.
- With BDE you can make hierarchical implementations allowing more visual clarity.
- You select modules from list of library modules or you create new ones with:
  - Primitive editor
  - FSM editor
  - BDE editor
Block Diagrams

Core modules

<table>
<thead>
<tr>
<th>Group</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arithmetic</td>
<td>BV^*</td>
</tr>
<tr>
<td></td>
<td>BV+</td>
</tr>
<tr>
<td></td>
<td>BV-</td>
</tr>
<tr>
<td></td>
<td>BV==</td>
</tr>
<tr>
<td></td>
<td>BV&lt;</td>
</tr>
<tr>
<td></td>
<td>BV&gt;</td>
</tr>
<tr>
<td>Comparison</td>
<td>Bitvector to Int</td>
</tr>
<tr>
<td></td>
<td>Int to Bitvector</td>
</tr>
<tr>
<td>Conversions</td>
<td>BVConstant</td>
</tr>
<tr>
<td>Number Generators</td>
<td>Uniform</td>
</tr>
<tr>
<td></td>
<td>PulseTrain</td>
</tr>
<tr>
<td>Traffic Generators</td>
<td>BV Create</td>
</tr>
<tr>
<td>Vector Operations</td>
<td>BV Mem Access</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Group</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALU_Arithmetic</td>
<td>Adder</td>
</tr>
<tr>
<td></td>
<td>Subtractor</td>
</tr>
<tr>
<td></td>
<td>ALU 32(fixed) functions</td>
</tr>
<tr>
<td>Clock Generators</td>
<td>Clock</td>
</tr>
<tr>
<td>Counters</td>
<td>Circular Down-Counter</td>
</tr>
<tr>
<td></td>
<td>Circular Up-Counter</td>
</tr>
<tr>
<td>Latch Registers</td>
<td>Latch Register</td>
</tr>
<tr>
<td>Shifters_BitOperations</td>
<td>And</td>
</tr>
<tr>
<td></td>
<td>Or</td>
</tr>
<tr>
<td></td>
<td>Nor</td>
</tr>
<tr>
<td></td>
<td>Left Shift by one</td>
</tr>
<tr>
<td></td>
<td>Right Shift by one</td>
</tr>
<tr>
<td>Memories</td>
<td>DRAM</td>
</tr>
<tr>
<td></td>
<td>FIFO</td>
</tr>
<tr>
<td></td>
<td>ROM</td>
</tr>
<tr>
<td></td>
<td>SRAM</td>
</tr>
<tr>
<td>Multipliers</td>
<td>Basic Multiplier</td>
</tr>
</tbody>
</table>
Core modules

<table>
<thead>
<tr>
<th>Group</th>
<th>Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS Access/Modify</td>
<td>Create DS</td>
</tr>
<tr>
<td></td>
<td>Insert Field</td>
</tr>
<tr>
<td></td>
<td>Select Field</td>
</tr>
<tr>
<td>Number Generators</td>
<td>Iconst</td>
</tr>
<tr>
<td></td>
<td>Rconst</td>
</tr>
<tr>
<td></td>
<td>TNOW</td>
</tr>
<tr>
<td></td>
<td>Random&gt;</td>
</tr>
<tr>
<td>Traffic Generators</td>
<td>Poisson Pulse Train</td>
</tr>
<tr>
<td>Execution Control</td>
<td>Execute in Order</td>
</tr>
<tr>
<td></td>
<td>Gate</td>
</tr>
<tr>
<td>Arithmetic</td>
<td>2 Input Expression</td>
</tr>
<tr>
<td>Loops</td>
<td>Int Do</td>
</tr>
<tr>
<td></td>
<td>Real Do</td>
</tr>
<tr>
<td>Delays</td>
<td>Abs Delay</td>
</tr>
<tr>
<td></td>
<td>Fixed Abs Delay</td>
</tr>
<tr>
<td>Quantity-Shared Resources</td>
<td>Allocate</td>
</tr>
<tr>
<td></td>
<td>Free</td>
</tr>
<tr>
<td>Server Resources</td>
<td>Service</td>
</tr>
</tbody>
</table>

- There is also a number of additional libraries for specific applications
  - MAC layer libraries
  - ATM libraries
  - TCP/IP libraries
  - Network layer libraries
  - ...

BONeS Blocks
Hierarchical implementation
Module Definition

ARGUMENTS

Implementation

INPUT PORTS

OR

OR

OUTPUT PORTS

Implementation of a module
Argument Types and Implementation

- Parameters
  - Parameter Argument
- Memories
  - Memory Argument
- Events
  - Event-Class Argument
- Resources
  - Quantity-Shared Resource
  - Server Resource

Datastructure editor (DSE)

- DSE is the tool used in creating new format for information traversing through network.
- Datastructures inherit fields from their parents so that
  - Parent DS can always accommodate its child.
  - Child cannot accommodate parent cause it cannot resolve all of the parameters for its fields from parent.
Datastructure

- DS consists fields with
  - Name
  - Type of information
  - Range of values
  - Default value
- Fields can be set types, meaning that individual values are unnumbered information.
- BONeS converts set types internally number information, needed in processing.

Simulation manager (SIM)

- Simulation manager is used in executing of simulations.
- Initial values for different parameters are chosen in simulation manager.
- Each case of simulation has to saved as session of its own.
Parameter setup in SIM

Interactive simulation manager (ISIM)

- ISIM is the tool used in debugging of system.
- ISIM can also be used to build animations out of system simulation.
- With ISIM one can view development of simulation through several steps and breakpoints.
Probes

- Probes are data collection points placed on ports and arguments
- Probe placement can be made anywhere in the model hierarchy
- Probe implementation can be block diagram, FSM, or C++ primitive

Postprocessor (PP)

- Postprocessor is a tool used to create plots from datasets of probes.
- Single plot can include
  - Several datasets from different probes.
  - Trend value from one probe of many iterations
**Project editor (PE)**

- Project editor is used to visualise development of project (system)
- PE can also be design canvas for blocks so that first only interfaces are created and after that implementation is done

---

**Definition of Dynamic Instantiation**

- Dynamic Instantiation (DI) lets a BONeS simulation create any number of identical instances of BONeS blocks
  - During the simulation, many instances of the block may exist simultaneously, but with varying creation and duration times
- During the simulation, each DI of a block behaves no differently than the conventional block and occupies the same location in the simulation data flow
Dynamic Instances

Advantages of Dynamic Instantiation

- Dynamic Instances help you solve modeling problems where objects are created, exist for some time and then are removed
- Examples
  - Mobile units entering, traversing, and exiting a coverage area
  - Creation and deletion of virtual circuits
  - Peer-to-peer protocol connections
IP: Reassembly Using Dynamic Instantiation

Packets from Network

Reassembly Block

Frame 0
0 0 0
Reassembly Block
DI Instance 0

Frame N
0 0 0
Reassembly Block
DI Instance N

Frame 1
0 0 0
Reassembly Block
DI Instance 1