# Status Report

**IBIS 4.1 Macro Working Group**

**IBIS Open Forum Summit**  
**July 25, 2006**

**presented by**  
Arpad Muranyi, Intel

---

## IBIS-Macro Working Group

<table>
<thead>
<tr>
<th>Intel - Arpad Muranyi</th>
<th>Cadence – Lance Wang, Ken Willis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cisco - Mike LaBonte, Todd Westerhoff</td>
<td>Mentor Graphics - Ian Dodd</td>
</tr>
<tr>
<td>IBM – Joe Abler</td>
<td>NC State University - Paul Fernando</td>
</tr>
<tr>
<td>Sigrity - Sam Chitwood</td>
<td>SiSoft - Barry Katz, Walter Kat</td>
</tr>
<tr>
<td>Teraspeed - Scott McMorrow, Bob Ross</td>
<td>TI – Richard Ward</td>
</tr>
</tbody>
</table>
Agenda

- IBIS-Macro group history
- IBIS-Macro library status
- Serial link design issues
- Phase 1 / Phase 2
- Requirements Definition
- Encryption
- Algorithmic Modeling
- AMS investigations
  - function library development
- API investigations
  - the evolution of IBIS
- Cadence proposal
- Current status
IBIS-Macro Group History

• Formed in 2005 to drive IBIS support for advanced device technologies

• Original goals:
  – leverage existing skills
    • most model developers are familiar with SPICE-style macro modeling
  – speed EDA/semiconductor adoption of advanced behavioral modeling techniques
  – multi-simulator support
    • same as original IBIS, tool-independent models
IBIS-Macro Library Status

• Developed library of standard building blocks in VHDL-AMS and Verilog-A
  – can be used in a SPICE style (netlist) fashion to build macro models for more complicated buffer models

• Developed automatic model translation utility
  – extracts data from IBIS file for use with the “IBIS buffer” building blocks of the library

• Created templates for several common model types

• Current released library version: 1.1
  – www.eda.org/pub/ibis/macromodel_wip/

• Further development waiting on testing and feedback from user community
Serial Link Design Issues

• Discussions with SERDES vendors revealed next generation devices require more complex models than possible with the current building block library:
  – receiver decision feedback equalization (DFE) circuitry
  – clock recovery (CDR) circuits and associated algorithms
  – complex driver models with arbitrary number of taps
• We decided complex SERDES devices required another level of modeling capability

  – Phase 1: existing building block library, suitable for modeling drivers with a small numbers of fixed taps

  – Phase 2: new strategy (TBD) for modeling multi-tap drivers, receiver DFE and CDR circuits
Requirements Definition

• Multi-EDA simulator support
• Multi-silicon vendor support
• Supports modeling at “algorithmic” level
• Reasonably compatible with silicon vendor SERDES design processes
  – ensures models will be timely and accurate
Encryption

• If driver/receiver algorithms are modeled and distributed, IP protection must be assured

• Encryption may be the only viable solution for protecting algorithm source code
  – are compiled code models safe enough to protect IP?

• Discussed EE Times article on Synplicity's open IP encryption scheme

• Discussed encryption related work in other workgroups
  – Accellera, IEEE

• Seems that there is an emerging solution which should be adopted by the IBIS Open Forum
Algorithmic Modeling

- DFE and CDR circuits are normally designed and validated at the “algorithmic” level by semiconductor vendors.
- In practice, the SERDES receiver input buffer separates the channel behavior from the receiver input circuit.
- Input signal processing can be thought of as a DSP case.
AMS Investigations

• Can AMS effectively be used to model DFE and CDR behavior?

• TI actively investigating with help from Gary Pratt of Mentor

• Will simulator performance be acceptable for the simulation lengths required?

• Will semiconductor vendors be willing to create AMS models?
  – compatibility with internal design libraries and methodologies using Matlab, C/C++ etc… is a major factor

• No conclusions yet
API Investigations

• Is there a need for an API, more flexibility in IBIS?
  – original IBIS assumes algorithms in tools (inflexible)
  – IBIS 4.1 adds languages (*.AMS) for flexibility (code your own algorithms), but there is no choice for other languages
  – API: any language allowed, connect simulator with compiled code

• Cadence proposed a “simulator API” mechanism during our June 19 and July 11 meetings

• API allows compiled model code to be linked into the simulator
  – does compiled code address IP protection issue?
Proposed Solution & Architecture

- Allow IC companies to develop “executable” algorithm based models that plug into the simulator through a dynamically linked library (dll)
- Simplest possible public API (C-wraper)
- Algorithmic Models in a dll
  - Can capture and encapsulate complex algorithms
  - Can add Jitter
  - Can include CDR modules
  - Protects IP without tool-specific encryption, no simulator specific encryption needed
  - Provides SERDES and EDA vendor independent interoperability if standardized
  - Can complete measurement loop – pluggable soft IP

EDA vendor

Rx DLL Libs

IC Co. IP

Waveform Processing

New Waveforms
Cadence Proposal

API Call Params

- long **rx_init** (double *a, long row_size, long col_size, double bitp, double tr, double tf, void **pdll_server_param_obj, void *dll_client_param, char *dllcontrol, [genchdllmsg_type **msg])
  - Input: Channel Characterization, system and dll specific parameters from config file
    - bit period, sampling intervals, # of forward/backward coefficients, …
  - Output: Modified Channel Characterization, status

- long **rx_getwave** (double *wave_in, long size, double dt, double *clk, void *dll_server_param_obj, void *dll_client_param, [genchdllmsg_type **msg])
  - Input: Voltage at Rx input at specific times
  - Output: Modified Voltage, Clock tics (dll specific), status

- long **rx_close** (void **ptr_2_dll_server_param_obj)
  - Clean up, exit

Note: items in [ ] are optional and can be 0(NULL)
Current Status

• We had a lot of discussion on encryption, but it seems that this will be taken care of by other workgroups

• We are currently discussing the API proposal
  – is it needed?
  – what should it include and look like
  – how could/should it be incorporated into the IBIS spec?
  – can we use [External Model] or [External Circuit] with modifications if necessary?
  – should we use other existing API interfaces, such as SystemC, etc…?

• Can higher level functions written in *-AMS in the macro library solve the problems?
  – a collection of functions similar to Matlab’s toolboxes
For More Information

• IBIS-Macro Website

• IBIS-Macro mail reflector
  – Mail to: ibis-macro-request@freelists.org
  – Subject: subscribe

• IBIS-Macro mail archives