Design Constraints Working Group
Kick-off Meeting

Mark Hahn - mhahn@cadence.com
Jin-sheng Shyr - shyrj@taec.com

March 16, 1998

Meeting Agenda

- Opening Statement, Self-introductions 1:00 (30 minutes)
  Who are we, and why are we here?
- Background 1:30 (45 minutes)
- Discussion: Nature of Design Constraints 2:15 (45 minutes)
- Break 3:00 (15 minutes)
- Discussion: Ways to Define Constraints 3:15 (30 minutes)
- Getting Started 3:45 (60 minutes)
  - Ways to Make the WG Successful
  - Meeting Logistics
  - Charter, Scope, Deliverables, Timeline
  - Topics for Investigation
- Wrap-up 4:45 (15 minutes)
  - Points of View Line-up
  - Action Items

Goals For Today's Meeting

- **Build a common knowledge base**
- **Share ideas and build consensus**
  - What to work on
  - What not to work on
  - What the end result should be
  - What is achievable, and when
  - Priorities
  - How to proceed
- **Prepare for subsequent meetings**

Background

- **Why develop a constraint standard?**
- **How would a constraint standard be used?**
- **Organizational framework**
  - Where does this WG fit in?
  - VSIA Relationship
- **Prior work**
  - Observations from SC-WG
- **Preliminary charter**

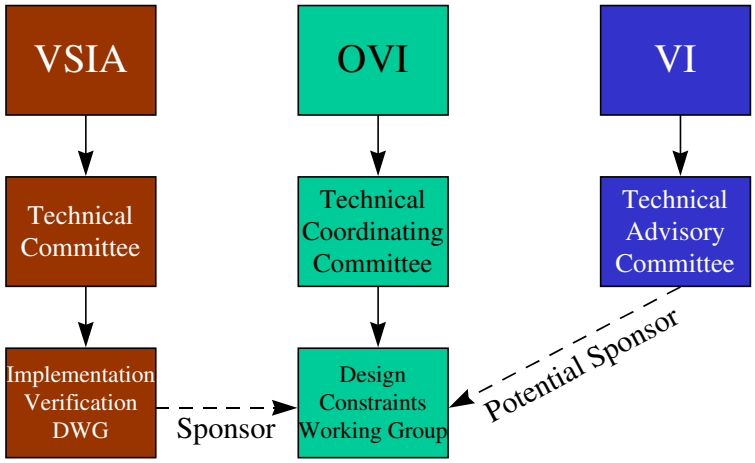
Why Develop A Constraint Standard?

- Tower of Babel today
  - Many different formats for describing constraints
  - Inconsistent syntax
    - Requires re-entering or translating constraints
  - Inconsistent semantics
    - May not be able to translate constraints
    - Contributes to lack of convergence
  - Wasted effort
    - Designers must spend significant time understanding what each tool supports and getting the constraints into each tool
    - EDA developers wind up defining new formats for each new tool
    - IP providers must supply the same data in multiple formats
    - IP integrators may have to translate internal constraints for IPs to get through their particular flow
    - Semiconductor vendors have a harder time qualifying tools

How Would A Constraint Standard Be Used?

- By designers
  - As a single, consistent way to describe their intent
- By EDA tool developers
  - As a standard way to read, write, and interpret constraints
- By IP providers
  - To describe their intent for partially implemented IP blocks
  - To describe restrictions on how IP blocks may be used
- By IP integrators
  - To complete the implementation of IP blocks
- By semiconductor vendors
  - As part of tool qualification
  - In creating design flows and kits

Where Does This Working Group Fit?



VSI	Virtual Socket Interface Alliance
OVI	Open Verilog International
VI	VHDL International

VSIA Relationship

- Formal Sponsorship: VSIA will
  - Recruit members
  - Provide requirements specifically for IP mix and match
  - Endorse the standardization effort
    - Based on commitment to address VSIA requirements
  - Review draft specification, provide feedback
  - Adopt the standard when approved
    - Provided it meets VSIA requirements
  - Promote the standard after approval

Prior Work

- Synthesis Constraints Working Group (SC-WG)
  - Formed in March, 1996 under OVI
  - Joint OVI/VI sponsorship in August, 1996
  - Charter
    - Synthesis tool interoperability
  - Focus
    - Definition of the General Constraint Language (GCL), a constraint command language for user entry
  - Problem
    - Consolidation of synthesis tools
  - Status
    - Fairly good progress on timing constraints
    - Inactive since October 1997
  - Details at http://www.vhdl.org/pub/scwg/index.html

Prior Work (2)

- General Constraint Format (GCF)
  - An exchange format for tool-to-tool communication
  - Cadence-proprietary format
    - Provided to SC-WG for review
  - Status
    - Initial emphasis on timing
    - Some area, power, parasitics constraints
    - Supported by many Cadence tools
    - Joint work with Ambit to write GCF
  - Plans
    - Continue to evolve to cover additional constraints
    - Consistent semantics with DC-WG command language
    - Possible standardization

Observations from SC-WG

- A constraint standard
  - Should express the designer's intent
    - Aspects of how the design should be implemented which aren't covered by functional descriptions
    - Aspects of the environment in which the design will operate
  - Should not describe tool-specific behavior or control flow
- Constraints need to be updated throughout the design flow
  - Designers change constraints based on progress in implementation
  - Tools generate additional constraints (budgeting, transformations)
  - Information about the tradeoffs between constraints changes as you go through the flow

Observations from SC-WG (2)

- Hierarchy is important
  - Many constraints apply to both logical and physical herarchy
    - Need to relate user specifications based on logical hierarchy to the physical hierarchy
  - Design object names may change as hierarchy is flattened or regrouped
  - Some tools are flat
    - Need to flatten hierarchical constraint descriptions

Observations from SC-WG (3)

- A constraint command language
  - Should
    - Define a set of constraint-related commands and their arguments
    - Be easy for a designer to enter
    - Provide powerful and expressive ways to specify which design objects are affected by a constraint
      - Macros, regular expressions
    - Allow constraint commands to be embedded in common extension languages, particularly Tcl
  - Should not
    - Be an extension language itself
      - No variables, looping, other programming language constructs
      - Avoid competition with tool-specific extension languages
  - Is relatively hard for tools to read

Observations from SC-WG (4)

- A constraint interchange format should
  - Define a set of constraint-related constructs
  - Be easy for tools to interpret and update
    - Provide limited ways to specify which design objects are affected by a constraint
  - Share semantics with a standard constraint command language

Preliminary Charter

- Original Proposal:
  - Define
    - A constraint command language (entered by users)
    - A corresponding interchange format (tool-to-tool)
  - Covering
    - Logic architecture
    - Timing
    - Area
    - Power
    - Test
    - Clocking
    - Physical Implementation
    - Environment/Operating Conditions
    - ...
  - Supporting language independence (Verilog, VHDL)

Nature of Design Constraints

Some thoughts to lead off discussion

- Major categories
  - High level design goals
    - Throughput, reliability, error rates, cost
  - Boundary conditions and operating environments
    - What is the environment around each block?
  - Budgets
    - Hierarchical partitioning of design goals
  - Special cases (exception handling)
  - Modes of operation
    - Don't care conditions
    - Mutually exclusive conditions
    - Infeasible states (false paths, feedback loops)
  - Detailed implementation controls

Nature of Design Constraints (2)

- Constraint Domains
  - Timing
  - Area
  - Power
  - Signal Integrity
  - Logic Architecture
  - Clocking
  - Test
  - Manufacturability
  - Reliability/Lifetime

Nature of Design Constraints (3)

- Different levels of abstraction
  - Many transformations of system design goals into detailed implementation constraints
  - Need ways to track derivation history
  - Low level constraints may be directly entered, rather than derived from higher level constraints
- Different levels of flexibility
  - Goals/objectives versus design rules
  - Tradeoffs between constraints
    - Smallest area which meets timing
    - Fastest design which meets power
- Multiple sources
  - System designers
  - Logic, physical designers (novice through expert)
  - IP providers
  - Cell library developers
  - Semiconductor vendors
  - Tools

Nature of Design Constraints (4)

- Multiple contexts
  - Constraint Entry/Management
  - Analysis
  - Estimation
  - Resource planning
  - Partitioning
  - Implementation
  - Optimization
    - Timing, area, power
  - Correction
    - Design rule violations
    - Signal integrity
  - Verification
  - Process Migration

Nature of Design Constraints (5)

- Multiple applications
  - Design Estimation
  - RTL Synthesis
  - Design Planning (floorplanning)
  - Timing Analysis
  - Timing-driven Layout
  - Timing (gate-level signoff) simulation
  - Post-layout or Location-based Optimization
  - Power Analysis

Ways to Define Constraints

Some thoughts to lead off discussion

- Command Language
  - Example: GCL
  - Mature methodology and compiler technology
  - Low cost way to achieve interoperability
  - Yet another language for designers to learn
  - Possible strategy: co-existence versus replacement

- Information Model and API
  - Example: DPCM
  - Object oriented, easy to evolve
  - Major investment in database, API, and programming language
  - Not for everyday designers

Ways to Define Constraints (2)

- Formal Specification
  - Example: EXPRESS
  - Top-down, applying theorem proving techniques
  - Hardest to implement
  - Not for every designer

- Attribute Dictionary
  - Designers can input in tabular form, like a spreadsheet
  - Tools to extract symbolic or physical values from the table
  - Not able to define semantics precisely
  - Not general enough to cover all conditions, 20-80 solution

Ways to Make the WG Successful

Some thoughts to lead off discussion

- Charter, Scope, Deliverables
  - Focus on expressing the designer's intent
  - Consider the design flow as a whole
  - Leverage previous work
    - Consider backward compatibility, but don't be driven by it
  - Avoid
    - Describing tool-specific behavior
    - Defining a mixture of extension language and constraints
    - A win-lose outcome
  - Add value to mix-and-match IP exchange

Ways to Make the WG Successful (2)

- Process
  - Begin with the end in mind
    - Start out with a paper pilot project
    - Define the roadmap from beginning to end
  - Break work into phases
    - Provide useful results early to build momentum
  - Break phases into sub-projects/sub-groups
    - Allow people to focus their time and effort on selected areas
  - Use the "Point of View" approach to build consensus

Ways to Make the WG Successful (3)

- Process (continued)
  - Prepare proposals off-line, circulate through e-mail
  - Use meetings to discuss proposals, not create them
  - Build high visibility
    - Press coverage
    - Presentations/tutorials at conferences
    - Endorsements
      - Companies, other standards organizations
    - Pilot projects

Getting Started

- Meeting Logistics
  - Bi-weekly teleconferences (2 hours)
    - What time?
      - Mornings are best for European, East Coast members
    - Can reduce time if enough work is being done off-line
    - VSIA will cover cost
  - Quarterly face-to-face meetings (1/2 day)
    - Coordinate with conferences to minimize travel
- Charter, Scope, Deliverables, Timeline
  - Focus on process for developing these
    - Not enough time to finalize today
    - Want to let people think about it

Getting Started (2)

- Topics for investigation
  - VI Sponsorship
    - Preliminary discussions with Gabe Moretti, Steve Schulz, Victor Berman
    - Technical Activities meeting this week
  - Constraint dictionary spreadsheet
  - Mixed signal constraints

Wrap-Up

- Points Of View

Wrap-Up (2)

- Action Items