Selected Languages and Formats

Paulo Flores

ESPRIT III Project # 6043 "QuickChips"
Task 1–5.4.4 - Deliverable 17

INESC
Instituto de Engenharia de Sistemas e Computadores

May 1994

Contents

1 Introduction

The capability to use tools form different vendors in the QuickChips methodology, makes possible the creation of a design environment tailored to the specific needs of a particular site. However, transferring design data between tools might be difficult, if an intermediate data format that the involved tools understand did not exist.

The use of standard languages and formats for the representation and transfer design data presents several benefits. Tool upgrading can be easily accomplished in a design environment using standard languages and formats, because they are supported by most commercial tools. It is not needed to be committed to a specific tool, if a higher performance or lower price tool became available. Cooperation between organizations during a project is simplified and the exchange of information at different levels of design process is possible. Since standards are usually of public domain, development of tools are also possible.

This report describes the set of languages and formats selected for use in the QuickChips design environment. A brief summary on the most important standards languages and formats in Electronic Design Automation (EDA) industry is presented in section 2. The selected languages and formats use in QuickChips design environment are stressed out by the data flow presented in section 3. Conclusions are presented in section 4.

2 Standard Languages and Formats

Standards can be imposed either by norms or by general usage. The former are developed by industry, government and/or academies and proposed to a standardization institute (such as IEEE or ANSI) to became a norm. The latter, are in general developed by industrial or commercial organizations that, after its release in public domain, become of popular use.

In the next sub-sections the most relevant standards in the EDA industry are presented.

2.1 VHDL - VHSIC Hardware Description Language

The VHDL is a standard language for the design and description of digital electronic circuits. The language development began in 1980 as the result of the VHSIC[1] project supported by the United States Department of Defense and included the participation of government, industry and university members.

The VHDL effort was initiated with the goal to address the issues of design complexity in a was that is independent of design implementation technology and process, is publicly available, and supports large scale design and reuse.

A team composed by Intermetrics, IBM, and Texas Instruments, developed version 7.2 of the language which was made public. In 1987, after incorporating some language recommendations from industry, academia and government, the IEEE certified VHDL as Standard 1076. The syntax and semantic of the standard are maintained by the VHDL Analyses and Standardization Group (VASG) of IEEE, and are documented in the VHDL Language Reference Manual [LRM 88]. The standard was rectified in 1992 [Bergé 93].

The VHDL language allows representations of circuits at differents levels of abstraction. A VHDL entity, which defines the interface of the system with its exterior, have its functionality described in an architecture using a combination of any of the following style/levels of description:

Structural Description - the design is represented as an arrangement of interconnected components. These components can be basic cells from the technology and/or other components also described in VHDL.

Dataflow Description - represents the behavior of the circuit in a more abstract way but with some structure implied. It is similar to a register transfer level (RTL) description.

Behavior Description - describes the behavior of the circuit without any structural information. This type of description is done using a sequence of programming language statements analogous to those used in algorithmic languages, but with a particular syntax and semantic.

Although VHDL started as a description and simulation language, currently it is also used in other domains, such as synthesis and verification. Due to its complexity, some

[1] Very High Speed Integrated Circuits

restrictions are usually made, and most of the tools support only a specific subset of the language.

To ensure the up-to-dateness of a standard, the IEEE requires its recertification every five years. So, a new version of the language, VHDL-92, was standardized in 1993 . This new version clarifies some syntax and semantics ambiguities and introduces some new features which improve the language use.

Being an independent standard language the VHDL is an attractive choice for circuit descriptions required to operate in a multi-vendor or multi-company design environment. For this reason many CAD tools accept it as way to describe a circuit.

2.2 Verilog

Verilog is a hardware description language originally designed in 1985 by Gateway Design Automation. This company was later merged with Cadence Design Systems and since then, the Verilog has been known and used as the language of the Cadence digital simulator Verilog-XL.

Verilog is based on the C programming language with some extensions to handle timing information. Like the VHDL language it allows modeling from gate level to behavioral level using descriptions in any of the three styles: structural, dataflow or behavioral.

In spite of being a proprietary language the use of Verilog was widely spread among IC designers (especially those using Cadence) and a large number of libraries and components models is available. Many foundries accept a Verilog simulation as a *sign off simulation* before going to production.

Later faced with the standardization of VHDL, Cadence decided to transfer Verilog to the public domain so it can be supported by any CAD tool. The maintenance and promotion of the language is now provided by Open Verilog International (OVI) which published in October 1991 the first version of the Verilog Hardware Description Language Reference Manual [Verilog 91].

2.3 EDIF - Electronic Design Interchange Format

The creation of a standard format which allows information exchange between CAD systems of different vendors was the main goal in the development of EDIF. The stan-

dardization process of the language began in 1983 and in 1987 version 2 0 0, which was lately adopted by ANSI (*American National Standards Institute*), became available.

The EDIF format represents the design data in a very structured form using a text description, with a Lisp like syntax. The design information is organized in libraries which are composed by cells, the main units of projects. Inside each cell the information is organized in views which describe the cells in different levels of abstraction. The views defined in EDIF represent the levels in which information exchange between CAD systems was expected:

- *NETLIST* - to transfer circuits as an interconnected list of components
- *SCHEMATIC* - to transfer the graphical representation of the circuit
- *MASKLAYOUT* - to transfer the physical layout of the circuit
- *SYMBOLICLAYOUT* - to transfer an abstract layout of the circuit
- *PCBLAYOUT* - to transfer printed circuit board information

The generalized use of EDIF presents some difficulties due to the wide range of representation levels defined and the existence of some syntax ambiguities in the language. For this reason a standard implementation does not exist, each vendor use a suitable subset of the language, which can lead to some incompatibilities. In spite of the existence of tools which are not able to read an EDIF file writen by themselves, the EDIF format serves the main goal of its development.

Currently, EDIF is the most used format for information transfer between CAD systems, particular for circuit transfer as an interconnected list of components (netlists views) or its graphical representation (schematics views). The main disadvantage in the use of this format is the sequential representation of information, which results in large text files. This disadvantage is critical in layout representation, where the quantity of information to describe the circuit is very large. For circuit description at this level other widely accepted formats exist, such as GDSII, which represents the circuit in a more compact way.

Recently new versions of EDIF were produced. These new releases provide improvements to the standard, by the elimination of some ambiguities, that will make easier the effort to encapsulate a tool in a framework. In addition they extend the range of EDIF

capability in response to user needs. The new releases, called EDIF Version 3 0 0, Version 4 0 0 and Version 5 0 offer: improved handling of connectivity and schematics (Version 3 0 0); much enhanced support for the representation of printed circuit board layout information (Version 4 0 0); and introduce a new test description capability into EDIF (Version 5 0 0).

2.4 GDSII

The GDSII format is a widely used industrial standard for the layout representation of a circuit. It is used as a standard interface format between design systems and manufacturing machinery.

As previous mentioned, this format allows a compact representation of the huge amount of geometric information necessary to describe all the layers used in the circuit fabrication. The compaction of information is achieved using a binary format of layout representation.

2.5 TSSI - Test Systems Strategies, Inc

The TSSI format in a host-independent and human-readable representation for simulation output files. This format was developed to fill the gap between the simulators or/and automatic test pattern generation (ATPG) and the tester machines.

The existence of converters from this format, provided by Test Systems Strategies, to most test machines, has turned this format as the industrial standard representation for test vectors.

The TSSI format is a tabular ASSCI format that contains state information about a set of signals monitored at all times where signal transitions occur. Each row has information about a test vector, using the state characters describe in table 1, and time instant it should be applied to the circuit. The mapping between each column and the signals name and type (input, output or bi-directional), are given in a separate signal definition file.

State Char	Meaning	State Char	Meaning
D	Force Low	X	Expect Unknown
U	Force High	T	Expect Hi-Z
N	Force Unknown	0	Unknown Direction Low
Z	Force Hi-Z	1	Unknown Direction High
L	Expect Low	?	Unknown Direction & Level
H	Expect High	F	Unknown Direction Hi-Z

Table 1: TSSI standard events format state characters

2.6 SDF - Standard Delay Format

The Standard Delay Format was developed by Cadence as a way to store timing information of an electronic system. This format became an industrial standard for transfer timing data between tools used at different stages in design process. Therefore, the data in SDF format is represented in a tool independent way, so that a layout tool can use design constraints identified during timing analysis or simulation tools can use the postlayout delay data.

With a lisp like syntax, the data in a SDF file can include:

- Delays: module path, device, interconnect, and port
- Timing checks: setup, hold, recovery, skew, width, and period
- Timing constraints: path and skew
- Incremental and absolute delays
- Conditional and unconditional module path delays and timing checks
- Design/instance-specific or type/library-specific data
- Scaling, environmental, technology, and user-defined parameters

The SDF format also supports hierarchical delay annotation. A design hierarchy might include several different ASICs (and/or cells or blocks within ASICs), each with its own SDF description.

3 Design Flow

The simplified data flow of figure 1, emphasizes the selected languages and formats used between the main project tasks of QuickChips design environment.

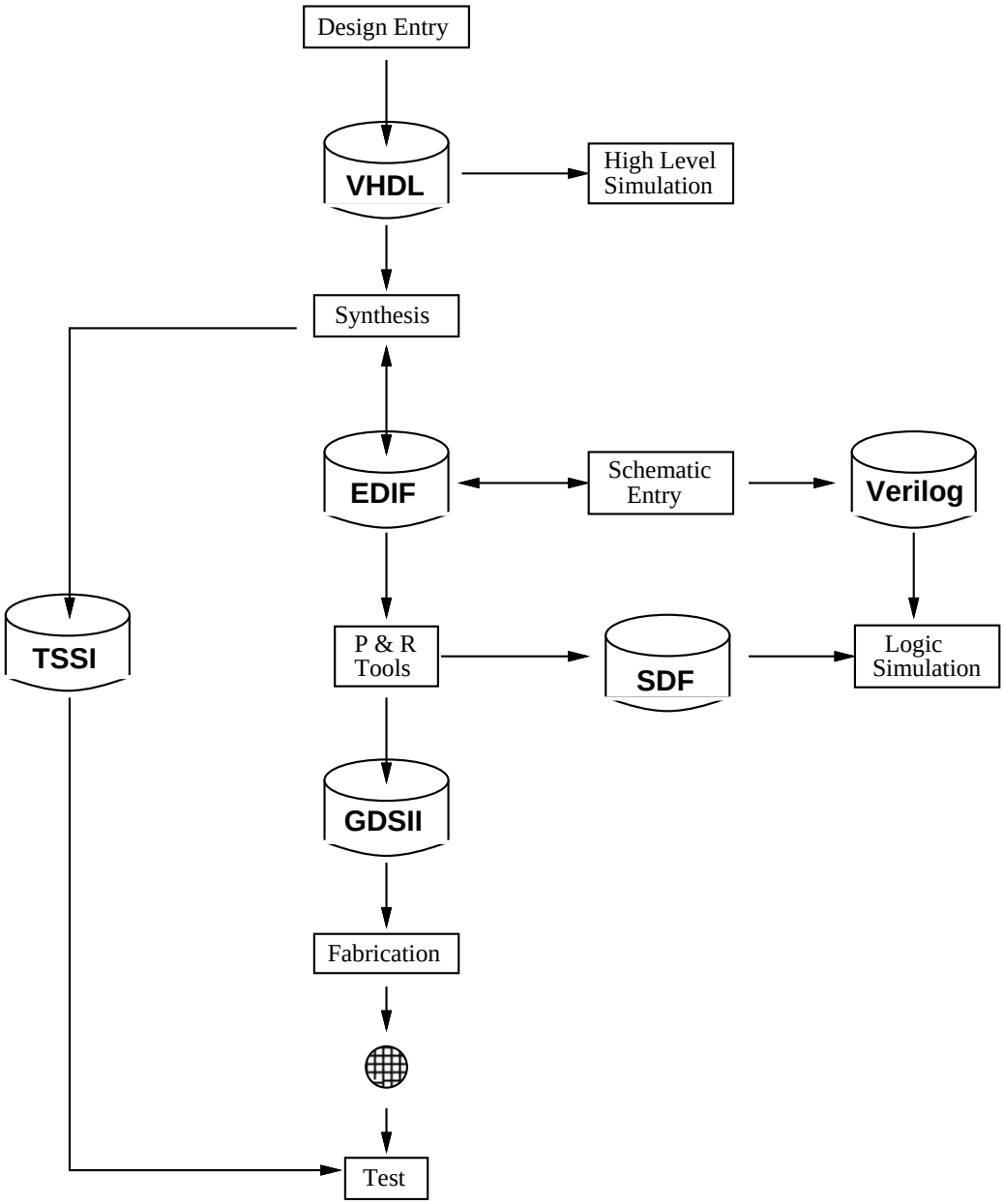

Figure 1: Data flow highlighting selected languages and formats.

The VHDL language is used for high level design representation. The code that describes the circuit can be generated automatically with the help of graphical tools or

edited manually by the designer. Once this description will be used by a synthesis tool, the designer must take into account some restrictions, as described in [Flores 93], in order to be assured that it can be synthesized and that the resulting circuit has the desired performance. The use of this standard language allows the use of any VHDL simulator to check the functionality of the description.

The EDIF format is used to transfer circuits descriptions between the synthesis and the schematic entry tools (NETLIST and/or SCHEMATIC views), and from any of these to the place and route tool (NETLIST view). The use of this standard format in QuickChips data flow allows to support low cost design entry tools that generate EDIF, such as PC-OrCAD.

Verilog is the language selected to describe a circuit, at a structural level, for the logic simulation. The use of Verilog is particularly suitable for logic simulation, due some of its capabilities, and benefits from the large number of existent libraries. Since Verilog can describe circuits at higher levels (dataflow and behavioral descriptions), it can also be used as a design entry language like VHDL (not represented in the figure).

The SDF format is used to backannotate the timing information of the circuit, after the place and route, for the logic simulator. This way it is possible to simulate the circuit with an accurate timing information, and check about functionality and performance.

The GDSII format is the used to transfer the circuit layout for production. It is generated by place and route tools and interpreted by the software of the DWL machine. A layout description in CIF[2] can also be produced for the propose of interfacing with others tools or machines.

The TSSI format is used as an interface between automatic pattern generation programs, ran during the synthesis task, and the tester machine. Test vectors may also be generated by a logic simulator that supports the TSSI standard (not represented in the figure).

4 Conclusions

The use of standard languages and formats in a design environment makes possible, and simplifies, the data transfer between tools of different vendors that are need for the design,

[2]Calthec Interchange Format - "old university standard" for layout descriptions

fabrication and test of a circuit. The existence of dialects or the use of differents languages subsets is a problem that can be overcome by in house development of "conversion" tools. This is only possible because standards definitions are in public domain.

The continuous evolution, and the emerging of new standards, are some of the issues that should be considered in the maintenance of a design environment and methodology. The new version of the VHDL language and EDIF format should be included in the design environment as soon as they become stable, and supported by commercial tools. Attention should also be given to "new standards" such as VITAL (VHDL Initiative Toward ASIC Libraries), which defines a style guide for the development of ASIC libraries in VHDL, or the synthesis package being developed by the Synthesis Working Group, so they can be supported by the QuickChips design methodology.

References

[Bergé 93] Jean-Michel Bergé, Alin Fonkoua, Serge Maginot, and Jacques Rouillard. *VHDL'92 - The New Features of the VHDL Hardware Description Language*. Kluwer Academic Publishers, 1993.

[Flores 93] Paulo Flores. Set of Synyhesis Tools. Technical report, Project ESPRIT 6043, November 1993.

[Harper 92] P. Harper and K. Scott. Towards A Standard VHDL Synthesis Package. In *Proceedings of European Design Automation Conference / Proceedings of Euro-VHDL*, September 1992.

[Khan 92] Hilary J. Khan and Richard F. Goldman. The Electronic Design Interchange Format EDIF: Present and Future. *Design Automation Conference*, pages 666–671, 1992.

[LRM 88] Institute of Electrical and Electronics Engineers. *IEEE Standard VHDL Language Reference Manual.*, March 1988. IEEE Std 1076-1987.

[Maginot 92] Serge Maginot. Evaluation Criteria of HDLs: VHDL compared to Verilog, UDL/I & M. *Proceedings of European Design Automation Conference / Euro-VHDL*, 1992.

[SDF 92] Cadence, Inc. *SDF Toolkit Reference Manual*, September 1992.

[TSSI 93] Test Systems Strategies. Inc. *TDS Software System - Master*, October 1993.

[Verilog 91] Open Verilog International. *Verilog Hardware Descript. Lang. Reference Manual*, October 1991. Version 1.0.

[VITAL 94] VITAL. *VHDL Initiative Toward ASIC Libraries - Model Development Specification*, March 1994. Version v2.2b.