INTERNATIONAL STANDARD

IEC 62265

First edition
2005-07

IEEE 1603™

Advanced Library Format (ALF) describing Integrated Circuit (IC) technology, cells and blocks

International Electrotechnical Commission, 3, rue de Varembé, PO Box 131, CH-1211 Geneva 20, Switzerland
Telephone: +41 22 919 02 11 Telefax: +41 22 919 03 00 E-mail: inmail@iec.ch Web: www.iec.ch

The Institute of Electrical and Electronics Engineers, Inc, 3 Park Avenue, New York, NY 10016-5997, USA
Telephone: +1 732 562 3800 Telefax: +1 732 562 1571 E-mail: stds-info@ieee.org Web: www.standards.ieee.org

Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

CONTENTS

INTERNATIONAL ELECTROTECHNICAL COMMISSION

ADVANCED LIBRARY FORMAT (ALF) DESCRIBING INTEGRATED CIRCUIT (IC) TECHNOLOGY, CELLS AND BLOCKS

FOREWORD

1) The International Electrotechnical Commission (IEC) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of IEC is to promote international co-operation on all questions concerning standardization in the electrical and electronic fields. To this end and in addition to other activities, IEC publishes International Standards, Technical Specifications, Technical Reports, Publicly Available Specifications (PAS) and Guides (hereafter referred to as "IEC Publication(s)"). Their preparation is entrusted to technical committees; any IEC National Committee interested in the subject dealt with may participate in this preparatory work. International, governmental and non-governmental organizations liaising with the IEC also participate in this preparation. IEC collaborates closely with the International Organization for Standardization (ISO) in accordance with conditions determined by agreement between the two organizations.
2) The formal decisions or agreements of IEC on technical matters express, as nearly as possible, an international consensus of opinion on the relevant subjects since each technical committee has representation from all interested IEC National Committees.
3) IEC Publications have the form of recommendations for international use and are accepted by IEC National Committees in that sense. While all reasonable efforts are made to ensure that the technical content of IEC Publications is accurate, IEC cannot be held responsible for the way in which they are used or for any misinterpretation by any end user.
4) In order to promote international uniformity, IEC National Committees undertake to apply IEC Publications transparently to the maximum extent possible in their national and regional publications. Any divergence between any IEC Publication and the corresponding national or regional publication shall be clearly indicated in the latter.
5) IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with an IEC Publication.
6) Attention is drawn to the possibility that some of the elements of this IEC Publication may be the subject of patent rights. IEC shall not be held responsible for identifying any or all such patent rights.

International Standard IEC/IEEE 62265 has been processed through IEC technical committee 93: Design automation.

The text of this standard is based on the following documents:

IEEE Std	FDIS	Report on voting
1603 (2003)	93/215/FDIS	93/221/RVD

Full information on the voting for the approval of this standard can be found in the report on voting indicated in the above table.

This publication has been drafted in accordance with the ISO/IEC Directives.

The committee has decided that the contents of this publication will remain unchanged until 2008.

IEC/IEEE Dual Logo International Standards

This Dual Logo International Standard is the result of an agreement between the IEC and the Institute of Electrical and Electronics Engineers, Inc. (IEEE). The original IEEE Standard was submitted to the IEC for consideration under the agreement, and the resulting IEC/IEEE Dual Logo International Standard has been published in accordance with the ISO/IEC Directives.

IEEE Standards documents are developed within the IEEE Societies and the Standards Coordinating Committees of the IEEE Standards Association (IEEE-SA) Standards Board. The IEEE develops its standards through a consensus development process, approved by the American National Standards Institute, which brings together volunteers representing varied viewpoints and interests to achieve the final product. Volunteers are not necessarily members of the Institute and serve without compensation. While the IEEE administers the process and establishes rules to promote fairness in the consensus development process, the IEEE does not independently evaluate, test, or verify the accuracy of any of the information contained in its standards.

Use of an IEC/IEEE Dual Logo International Standard is wholly voluntary. The IEC and IEEE disclaim liability for any personal injury, property or other damage, of any nature whatsoever, whether special, indirect, consequential, or compensatory, directly or indirectly resulting from the publication, use of, or reliance upon this, or any other IEC or IEEE Standard document.

The IEC and IEEE do not warrant or represent the accuracy or content of the material contained herein, and expressly disclaim any express or implied warranty, including any implied warranty of merchantability or fitness for a specific purpose, or that the use of the material contained herein is free from patent infringement. IEC/IEEE Dual Logo International Standards documents are supplied “AS IS”.

The existence of an IEC/IEEE Dual Logo International Standard does not imply that there are no other ways to produce, test, measure, purchase, market, or provide other goods and services related to the scope of the IEC/IEEE Dual Logo International Standard. Furthermore, the viewpoint expressed at the time a standard is approved and issued is subject to change brought about through developments in the state of the art and comments received from users of the standard.

Every IEEE Standard is subjected to review at least every five years for revision or reaffirmation. When a document is more than five years old and has not been reaffirmed, it is reasonable to conclude that its contents, although still of some value, do not wholly reflect the present state of the art. Users are cautioned to check to determine that they have the latest edition of any IEEE Standard.

In publishing and making this document available, the IEC and IEEE are not suggesting or rendering professional or other services for, or on behalf of, any person or entity. Neither the IEC nor IEEE is undertaking to perform any duty owed by any other person or entity to another. Any person utilizing this, and any other IEC/IEEE Dual Logo International Standards or IEEE Standards document, should rely upon the advice of a competent professional in determining the exercise of reasonable care in any given circumstances.

Interpretations – Occasionally questions may arise regarding the meaning of portions of standards as they relate to specific applications. When the need for interpretations is brought to the attention of IEEE, the Institute will initiate action to prepare appropriate responses. Since IEEE Standards represent a consensus of concerned interests, it is important to ensure that any interpretation has also received the concurrence of a balance of interests. For this reason, IEEE and the members of its societies and Standards Coordinating Committees are not able to provide an instant response to interpretation requests except in those cases where the matter has previously received formal consideration.

Comments for revision of IEC/IEEE Dual Logo International Standards are welcome from any interested party, regardless of membership affiliation with the IEC or IEEE. Suggestions for changes in documents should be in the form of a proposed change of text, together with appropriate supporting comments. Comments on standards and requests for interpretations should be addressed to:

Secretary, IEEE-SA Standards Board, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA and/or General Secretary, IEC, 3, rue de Varembé, PO Box 131, 1211 Geneva 20, Switzerland.

Authorization to photocopy portions of any individual standard for internal or personal use is granted by the Institute of Electrical and Electronics Engineers, Inc., provided that the appropriate fee is paid to Copyright Clearance Center. To arrange for payment of licensing fee, please contact Copyright Clearance Center, Customer Service, 222 Rosewood Drive, Danvers, MA 01923 USA; +1 978 750 8400. Permission to photocopy portions of any individual standard for educational classroom use can also be obtained through the Copyright Clearance Center.

NOTE – Attention is called to the possibility that implementation of this standard may require use of subject matter covered by patent rights. By publication of this standard, no position is taken with respect to the existence or validity of any patent rights in connection therewith. The IEEE shall not be responsible for identifying patents for which a license may be required by an IEEE standard or for conducting inquiries into the legal validity or scope of those patents that are brought to its attention.

IEEE Standard for an Advanced Library Format (ALF) Describing Integrated Circuit (IC) Technology, Cells, and Blocks

Sponsor

Design Automation Standards Committee
of the
IEEE Computer Society

Approved 11 September 2003

IEEE-SA Standards Board

Approved 29 December 2003

American National Standards Institute

Abstract: ALF is a modeling language for library elements used in IC technology. ALF enables description of electrical, functional, and physical models in a formal language suitable for electronic design automation (EDA) application tools targeted for design and analysis of an IC. This standard provides rules that describe ALF and how tool developers, integrators, library creators, and library users should use it.
Keywords: behavioral, block, cell, derate, EDA, electrical, format, functional, gate-level, integrated circuit, language, layout, library, modeling, physical, power, RTL, signal integrity, technology, timing

IEEE Introduction

The purpose of ALF is to provide a modeling language and semantics for the functional, physical, and electrical performance description of technology-specific libraries for cell-based and block-based design. Without a standard, EDA tools would be left to use tool-specific and fragmented library descriptions. The semantics would be defined by tool implementations only, which are subject to change and prone to misinterpretation. Therefore, ALF is proposed to create a consistent library view suitable as a reference for library creators and users, as well as for electronic design automation (EDA) tool developers and integrators.

IEEE Std 1603-2003 is based on the work of Open Verilog International (OVI) and its successor organization, Accellera.

The ALF standard began as the OVI Power & Synthesis Technical Steering Committee (PS-TSC) early in 1996, with the charter to define a standard library data format for synthesis, power analysis, and optimization. As the committee grew in membership, with the addition of experts in other fields such as design for test, it became clear that such a format could be easily extended to cover other design tools. Furthermore, the benefit to both silicon and EDA vendors of having a single, flexible format that would fully describe the functional, electrical, and physical performance of a technology library in an accurate and unambiguous fashion was widely recognized.

ALF was announced at the occasion of the OVI/VI-sponsored HDL conference in March 1997, where a trial version of the standard was released. Among the pioneers of proving the feasibility of ALF was the European CAD Standardization Initiative, sister organization of VSIA, who demonstrated an ALF-based ASIC implementation flow in 1997. In November 1997, OVI approved and released ALF version 1.0.

In 1998, the ASIC Council, under the auspices of the Silicon Integration Initiative (SI2), selected ALF as a complementary description of library elements within the open library architecture (OLA), which builds upon the IEEE 1481™-1999 standard for a delay calculation system. This endorsement triggered the initial adoption of ALF libraries by major ASIC vendors and the development of ALF version 1.1, which was approved and released by OVI in April 1999.

In June 1999, the ASIC council encouraged the ALF workgroup to include layout modeling. Consequently, deep submicron (DSM) issues, such as on-chip interconnect modeling, signal integrity, and reliability, became a major focus for ALF. The work culminated in the release of ALF version 2.0 in December 2000, under the auspices of the OVI/VI successor organization Accellera.

ALF version 2.0 became the foundation for this IEEE standard. An IEEE study group was formed in February 2001. The study group became the IEEE P1603 Working Group in June 2001. The name ALF has been retained due to already existing name recognition. By that time, the ALF had already set a standard for the industry, which can be measured by direct adoption and the influence on existing vendor-proprietary library formats. Major EDA vendors also made the specification of their existing proprietary library formats available to the industry and allowed the user community to extend those formats and strive for compatibility with ALF.

Although IEEE is now the legal owner of ALF, Accellera continues to foster and promote ALF. As a result, ALF has gained the attention of other national and international standardization bodies, such as JEITA in May 2002 and the IEC in October 2003.

From its inception, the goal for ALF has been to provide a solid foundation for library modeling within a continuously evolving application space. ALF has been designed to be more general in scope and purpose than a particular tool-oriented format. At the same time, care has been taken to make ALF easily adoptable

and to make the migration path from legacy formats as smooth as possible. Therefore, an ALF library can be very similar in appearance to a library in a conventional format, but ALF also has the expression power of a modeling language.

The construction principles for ALF can be summarized as follows:

— *Simplicity.* ALF has relatively few basic syntax construction principles. Once they are understood, reading and writing an ALF library or translating other library formats into ALF is very easy. Also, attention has been paid to the fact all ALF keywords are taken from natural language, i.e., spoken and written English, and their semantic meaning is as close to the natural language as possible. The use of artificial words or acronyms is limited to constructs, which have already become part of technical language in the industry.

— *Completeness.* Conventional library formats would support data without self-evident meaning, such as coefficients, scaling factors, etc. The interpretation of the data would be left to the application tool. On the other hand, an ALF library specifies a complete and self-contained description by providing the complete model, i.e., a calculation rule using an *arithmetic expression*. Furthermore, ALF contains information for characterization of particular measurement data, for example, delay, power, or noise. ALF introduces the original concept of a *vector expression* to describe the event pattern associated with the measurement. This concept has a far-reaching potential for creating abstract, yet accurate, modeling views for cells and larger blocks. Any timing, power, or signal integrity measurement on a digital circuit or a mixed-signal circuit can be associated with a vector expression.

— *Orthogonality.* Orthogonality allows for modeling features to be combined most efficiently with each other to yield a maximum expression capability. In ALF, orthogonality is closely related to context-sensitivity. A particular semantic meaning is created by describing a particular model in a particular context. For example, a model for capacitance can be described in the context of a wire, pin, or rule. A model for delay can be described in the context of a cell or wire. In a nonorthogonal approach, different keywords might be used for cell delay, wire delay, etc., and the fundamental semantics of delay would not be inherited by each construct.

— *Reusability and self-extensibility.* ALF supports the language constructs *template* and *group*, which allow for efficient representation of replicated statements with parameterized values. As these constructs follow the principle of orthogonality, a template can be used for parameterizing any ALF statement and not just a particular type of statement, such as a lookup table. ALF also supports language constructs for the definition of new keywords, their usage for construction of statements, and the context where they can be used.

In summary, ALF is a well-structured language that supports a true superset of virtually all existing library formats. Its conciseness and unique description features make it well-suited for innovative EDA applications.

ADVANCED LIBRARY FORMAT (ALF) DESCRIBING INTEGRATED CIRCUIT (IC) TECHNOLOGY, CELLS AND BLOCKS

1. Overview

This clause explains the scope and purpose of this standard, gives an overview of its applications, explains the conventions used, and summarizes its contents.

1.1 Scope and purpose

The scope of this standard is to serve as the data specification language of library elements for design applications used to implement an integrated circuit (IC). The range of abstraction shall include from the register-transfer level (RTL) to the physical level. The language shall model behavior, timing, power, signal integrity, physical abstraction and physical implementation rules of library elements.

Library elements for implementation of an IC include sets of predefined components, composed of transistors and interconnect, and sets of predefined rules for the assembly of such components. The design of application-specific ICs (ASICs) in particular relies on the availability of predefined components, called cells. An IC that uses large predefined compound library elements with a standardized functionality, for example, microprocessors as building blocks, is called a system on a chip (SOC).

The design of an ASIC or an SOC involves electronic design automation (EDA) tools. These tools assist the designer in the choice and assembly of library elements for creating and implementing the IC and verifying the functionality and performance specification of the IC. In order to create an IC involving several million instances of library elements within a manageable time period counted in weeks or months, the usage of EDA tools is mandatory.

A suitable description of library elements for design applications involving EDA tools is required. A key feature is to represent a library element at a level of abstraction that does not reveal the implementation of the library element itself. This is important for the following reasons:

— The complexity of the design data itself mandates data reduction.

— The complexity of the verification process, i.e., the verification for functional, physical, and electrical correctness, mandates that a library element is already characterized and verified by itself. Only the data necessary for creation and verification of the assembled IC is represented in the library.

— A library element is considered an intellectual property (IP) of the library provider.

Therefore, the purpose of this standard is to provide a modeling language and semantics for the functional, physical, and electrical performance description of technology-specific libraries for cell-based and block-based design. Without a standard, EDA tools would use multiple proprietary and tool-specific library descriptions. The semantics would be defined by tool implementations only, which are subject to change and prone to misinterpretation. Also, there would be redundancy using multiple descriptions for similar library aspects. Therefore, this standard proposes to create a consistent library view suitable as a reference for IC designers as well as for EDA tool developers and integrators.

1.2 Application of this standard

The ALF standard can be used in many different places throughout the design flow. The major uses include creation and characterization of library elements, basic implementation and performance analysis of an IC, and hierarchical implementation and virtual prototyping of an IC.

An application, as described in 1.2.1, 1.2.2, and 1.2.3, shall be called *compliant to ALF*, if and only if it satisfies the following criteria:

a) An application tool that uses ALF as input is capable of parsing any ALF file according to the rules specified in Clause 5, Clause 6, Clause 7, Clause 8, Clause 9, and Clause 10, even if not all data in that file is used by the application. In this way, one ALF library can be used for multiple applications with different scope.

b) A tool, as referred to in item a), uses a well-defined set of data from the ALF file within the scope of its application and interprets this data according to the rules specified in Clause 5, Clause 6, Clause 7, Clause 8, Clause 9, and Clause 10. In this way, any two applications using the same set of ALF data interpret the ALF data in the same consistent way.

c) An application tool that uses ALF as output is capable of generating an ALF file according to the rules specified in Clause 5, Clause 6, Clause 7, Clause 8, Clause 9, and Clause 10, and the generated file contains a well-defined set of data for an application as referred to in item a).

The following conventions are used in the flow diagrams depicted in Figure 1, Figure 2, Figure 3, and Figure 4:

— *Rectangle*: data file, format optionally indicated in parentheses

— *Oval*: application

— *Solid arrow*: existing, established function in the design flow

— *Dotted arrow*: possible design flow

1.2.1 Creation and characterization of library elements

ALF can be used to specify the desired functionality and characterization space of a library element, i.e., a cell.

The application for creation of a cell is shown in Figure 1.

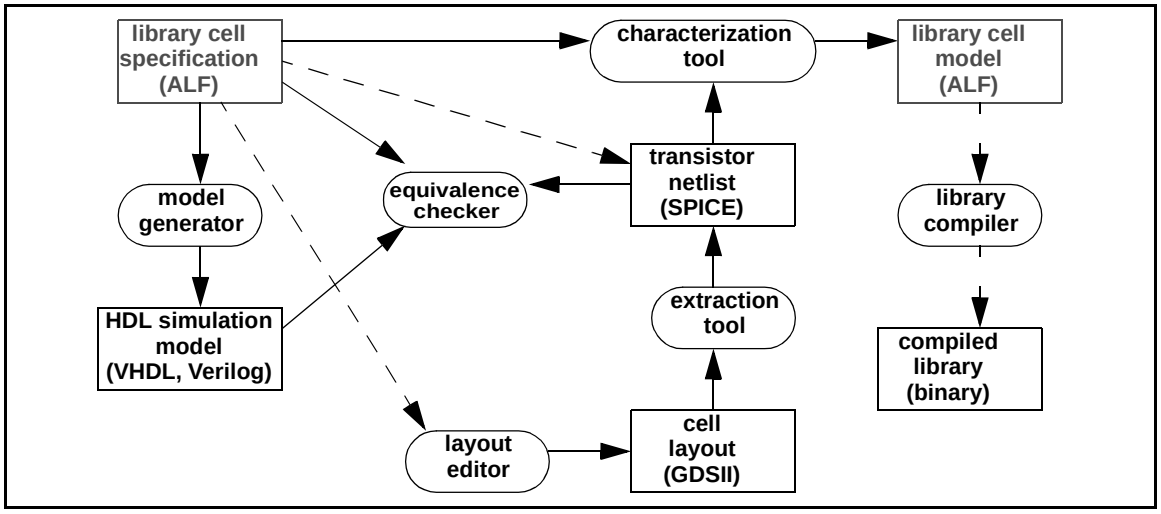

Figure 1—Cell library creation flow

A specification of a library element, i.e., a cell (see 8.2, 8.4), can be described in ALF. This specification includes the name of the cell and its terminals, i.e., pins (see 8.6), and a formal description of the function (see 9.1) performed by the cell. This formal description is sufficient for the purpose of generating hardware description language (HDL) simulation models in various languages, for example, VHDL (see IEEE Std 1076™-2002)[1] or Verilog (see IEEE Std 1364™-2001).

Multiple HDL models can be generated for different purposes, where the difference is defined by the user's preference for modeling style rather than by the functionality of the cell. For example, one model can handle unknown logic states in a crude way, resulting in fast simulation, while another model can handle unknown logic states in a case-by-case way, resulting is slow but more accurate simulation. The ALF model can serve as a common reference for all those HDL models.

A physical layout of a cell can be represented in the GDSII format. A transistor-level netlist of a cell in SPICE format (*SPICE 2G6 User's Guide* [B7][2]) can be extracted from the physical layout. Such a transistor netlist includes parasitic electrical components. Alternatively, a designer can create a transistor netlist by hand or by using an EDA tool that maps a functional specification described in ALF into a transistor-level netlist. Such a transistor netlist is less accurate than one extracted from layout, but can still be useful for prototyping a library.

Both the transistor netlist and the various HDL models can be compared against the functional specification described in ALF. More importantly, the transistor netlist can be used to characterize the performance of the cell, i.e., measure timing, power, noise (see 10.11), and other electrical characteristics (see 10.15) by running a SPICE simulation. The set of necessary SPICE simulations is determined and controlled by a characterization tool. The characterization tool can infer pertinent information from the specification represented in ALF, as far as this information relates to the functionality of the cell itself. For example, the timing arcs that need to be characterized can be represented in or inferred from ALF. The output of the characterization tool is a library cell model, populated with characterization data, also represented in ALF.

Optionally, a library compiler can be used to combine all the library cell models into a binary file, as a data preparation step for an EDA application tool.

[1]For information on references, see Clause 2.

[2]The numbers in brackets correspond to those in the bibliography in Annex D.

1.2.2 Basic implementation and performance analysis of an IC

The ALF library can be used in an IC implementation flow which uses cells as building blocks, in particular, an ASIC implementation flow.

A basic flow for an IC implementation using cells as building blocks is shown in Figure 2.

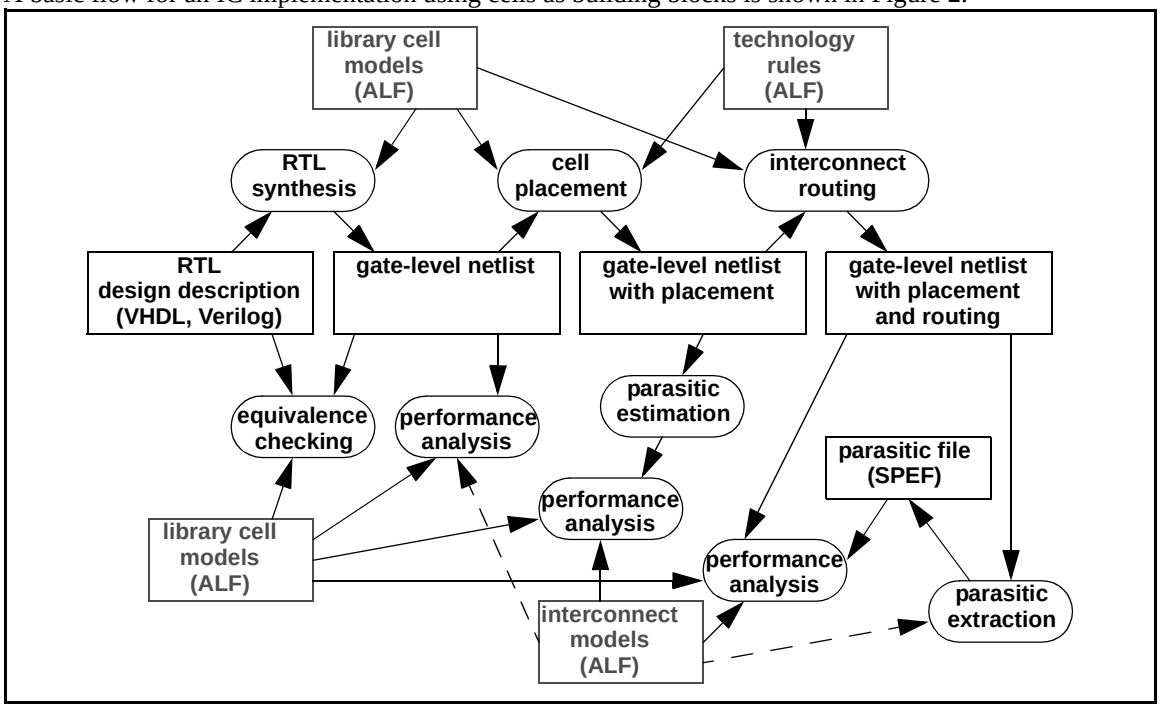

Figure 2—Basic IC implementation flow

In this flow, an RTL design description is transformed into a netlist by an RTL synthesis tool. The netlist contains instances of cells, also called gates, rather than transistors. This application can use the ALF library to find the library elements needed to map the RTL description into a netlist containing instances of cells. The transistors inside the cells are not described in the ALF cell models.

An equivalence checking tool can be used to decide whether the RTL-to-netlist transformation has been done correctly, by comparing the RTL design description with the netlist. This application can use the same ALF library as the RTL synthesis tool. Also, an HDL simulation tool (not shown in Figure 2) can be used to decide whether both the RTL design description and the netlist behave as expected in response to a given stimulus. The simulation tool can use an ALF model or an HDL model derived from the ALF model (see 9.4, 9.6).

The flow in Figure 2 is simplified. Special netlist transformations, such as the creation of data path structures, creation structures related to design for test (DFT), and especially scan insertion, are not shown here. However, the ALF cell models also contain information pertaining to these applications (see 8.5.3, 8.5.5, 8.5.6, 8.8.12, 9.2, 9.7).

The process of cell placement and interconnect routing is summarily referred to as layout. Special layout operations, such as the layout of a power supply structure or of a clock network structure, are not explicitly shown in Figure 2. The ALF cell models contain abstract physical information, such as the size and shape of the cell, and the location, size, and shape of the cell pins and routing blockages, which are pertinent for layout (see 8.22, 8.23). Also, abstract information concerning the artwork within the cell can be represented in ALF, for example, the area, perimeter, and connectivity of artwork on specific layers (see 10.18, 10.19).

This information is pertinent for manufacturability, such as antenna rule (see 8.21, 10.19.1, 10.19.2, 10.19.3) and metal density checks (see 8.31, 10.19.11).

In addition to cell models, technology rules for routing can also be represented in ALF, such as constraints for the width and length of routing segments, the distance between routing segments, the distance between vias, etc. (see 8.16, 8.18, 8.20, 10.19.7, 10.19.8, 10.19.9).

The implemented IC needs not only be correct in terms of functionality and layout, it also has to meet electrical performance constraints, predominantly timing constraints. Other aspects of electrical performance, such as power consumption, signal integrity, and reliability, have become increasingly important. Signal integrity aspects include the cleanliness of signal waveform shapes and the immunity against noise induced by crosstalk and voltage drop (see 10.11.1, 10.11.14, 10.15.1, 10.15.2). Reliability aspects include dependable long-term operation in the presence of electromigration stress, hot electron effect, and thermal instability. The cell models in ALF support characterization data for timing, power, signal integrity, and reliability. For example, reliability data can be described as a limit for voltage, current, or operation frequency (see 10.11.2). A particular feature in ALF is the representation of these data in the context of a stimulus, described by a *vector expression* (see 8.14, 9.12, 9.13). With this feature, the data can be related to particular environmental operation conditions, and a more accurate performance analysis can be performed.

Performance analysis happens within each step of the IC implementation process. RTL synthesis, cell placement, and interconnect routing applications have embedded static timing analysis (STA) and other performance analysis capabilities. Also, after completion of each step, a stand-alone performance analysis can be applied to measure the achieved performance more accurately.

Electrical performance depends not only on the interaction between instances of cells, but also on the parasitics introduced by the interconnect wires. After netlist creation, parasitics can be statistically estimated using a wire load model (WLM). After placement, parasitics can be more accurately predicted by estimating the length of particular routing wires between pins of placed cells. After routing, actual parasitics can be extracted and represented in a file using the standard parasitic exchange format (SPEF) (IEEE Std 1481™-1999 [B4]). An interconnect model in ALF can describe a statistical WLM, a rule for parasitic estimation based on estimated routes, or an interconnect analysis model (see 8.10, 8.12). The interconnect analysis model specifies the desired level of granularity for the parasitics (see 10.15.3, 10.15.4, 10.15.5, 10.16) and the calculation of timing, noise, voltage, or current based on instances of parasitics and on an electrical model of a driver cell. The data for the electrical model of a particular driver cell can be represented in ALF as a part of the cell characterization data.

1.2.3 Hierarchical implementation and virtual prototyping of an IC

An IC implementation flow with cells as building blocks has its limits imposed by the number of objects, i.e., the instances of cells and nets that can be reasonably handled by designers and by application flows.

For ICs exceeding the limits of objects that can be reasonably handled, the following approaches are used, possibly in combination with each other:

— *Bottom-up design*: Create larger building blocks from cells first, then use these blocks for IC implementation.
— *Top-down design*: Divide a design into subdesigns first, implement each subdesign as a block, then assemble the blocks.
— *Virtual prototyping*: Do a simplified so-called virtual implementation of the entire design first, then partition the virtually implemented design into blocks, use the results of the virtual implementation as constraints for actual implementation of each block, and implement and assemble the blocks.

The common denominator for all these methods is creation of blocks, in order to reduce the number of objects seen by the application.

The application for creation of a block is shown in Figure 3.

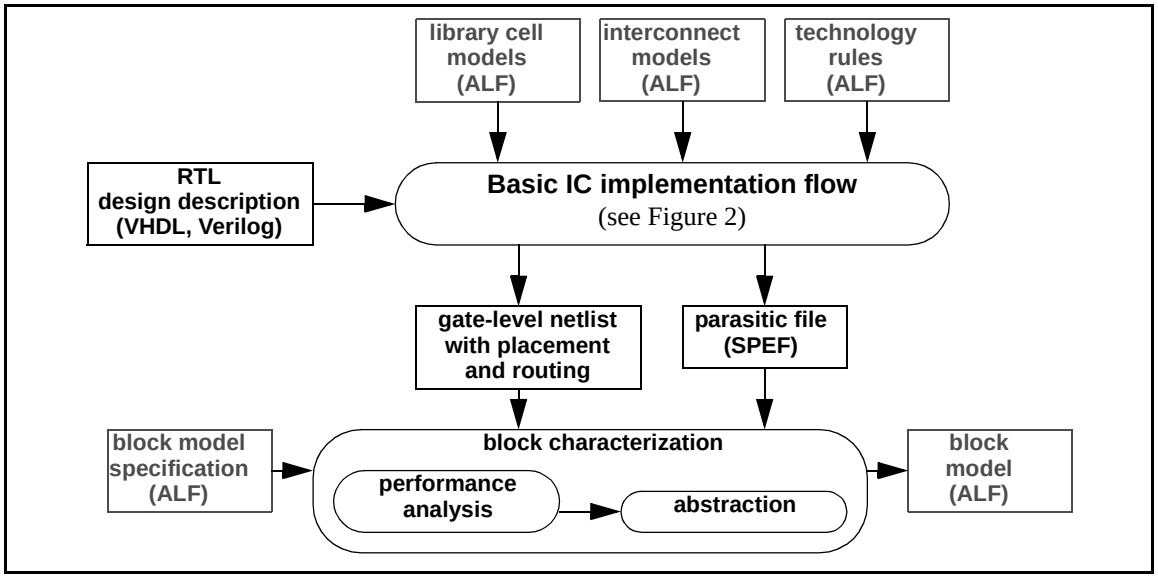

Figure 3—Block creation flow

A block can be created by using the basic IC implementation flow (see Figure 2). A block with a functionality that can be used and reused is commonly referred to as IP of the designer. In case of a “hard” block, the primary output of the implementation flow, i.e., a gate-level netlist with placement and routing, is preserved and eventually transformed into a physical artwork. In case of a “soft” block, only the primary input of the implementation flow, i.e., the RTL design description, is preserved. The output of the implementation flow serves only for the purpose of block characterization, i.e., creation of an abstract model for the block. The block characterization consists of a repeated application of performance analysis within the range of desired characterization followed by abstraction. Abstraction includes reduction of the physical implementation data and association of the performance analysis data with a specified model. Both the specification of the model and the model itself can be represented in ALF.

Variants to this flow include partial IC implementation, for example, only RTL synthesis and placement without routing, especially in the case of a soft block, where the implementation data is not preserved. The rationale for not preserving the implementation data of a block is the possibility of achieving a better overall IC implementation result by implementing the block later in the context of other blocks, instead of implementing the block stand-alone up front.

Depending on whether a block is used as a hard block or a soft block, the ALF model can represent a different level of abstraction. An ALF model for a hard block can have features similar to an ALF model for a cell (see 1.2.1 and 1.2.2). In addition, the netlist and the parasitics representing the output of the implementation flow can be partially preserved in the ALF model, especially at the boundary of the block (see 9.5). This enables accurate analysis of the electrical interaction of a block with adjacent blocks in the context of an IC implementation. On the other hand, an ALF model for a soft block can represent a statistical range or upper and lower bounds (see 10.5) for characterization data rather than “hard” characterization data, since there is a degree of variability in the implementation of actual instances of the block. Also, a statistical WLM can be encapsulated within the model of the block.

ALF supports specific modeling features for parameterizeable blocks, i.e., blocks that can be implemented in various physical shapes or sizes and with variable bitwidth and performance characteristics. The ALF

constructs *group* (see 7.14), *template* (see 7.15), *static,* and *dynamic template instantiation* (see 7.16) can be used for this purpose.

Independent of whether a block is a hard block or a soft block, the application for creating the IC can now use the abstract model of the block as a library element rather than using a cell. In a similar way, as an ALF model of a cell does not reveal transistor-level implementation details, an ALF model of a block does not reveal gate-level implementation details. However, the ALF model of a block still provides enough information for an application to implement or explore the implementation of an IC and analyze the performance and the compliance to logical and physical design constraints.

An IC is designed in the context of a specific environment with specific constraints. Environmental constraints include for the characteristics of the package, the printed board, the range of process, voltage, and temperature (PVT) conditions (see 10.14). Other constraints are given by globally applicable physical design rules, for example, the available routing layers, the amount of routing resources reserved for the power distribution, and the available locations for IO pins at the boundary and in the center of a chip. The virtual prototyping approach can be used to evaluate whether a design can be implemented within these constraints. The electrical characterization data in ALF, i.e., timing, power, noise, physical and electrical rules, estimation models for parasitics, etc., can be represented as mathematical functions of environmental conditions and constraints (see 10.3, 10.4).

A conceptual flow for the virtual prototyping and hierarchical implementation of an IC involving ALF models at different levels of abstraction is shown in Figure 4.

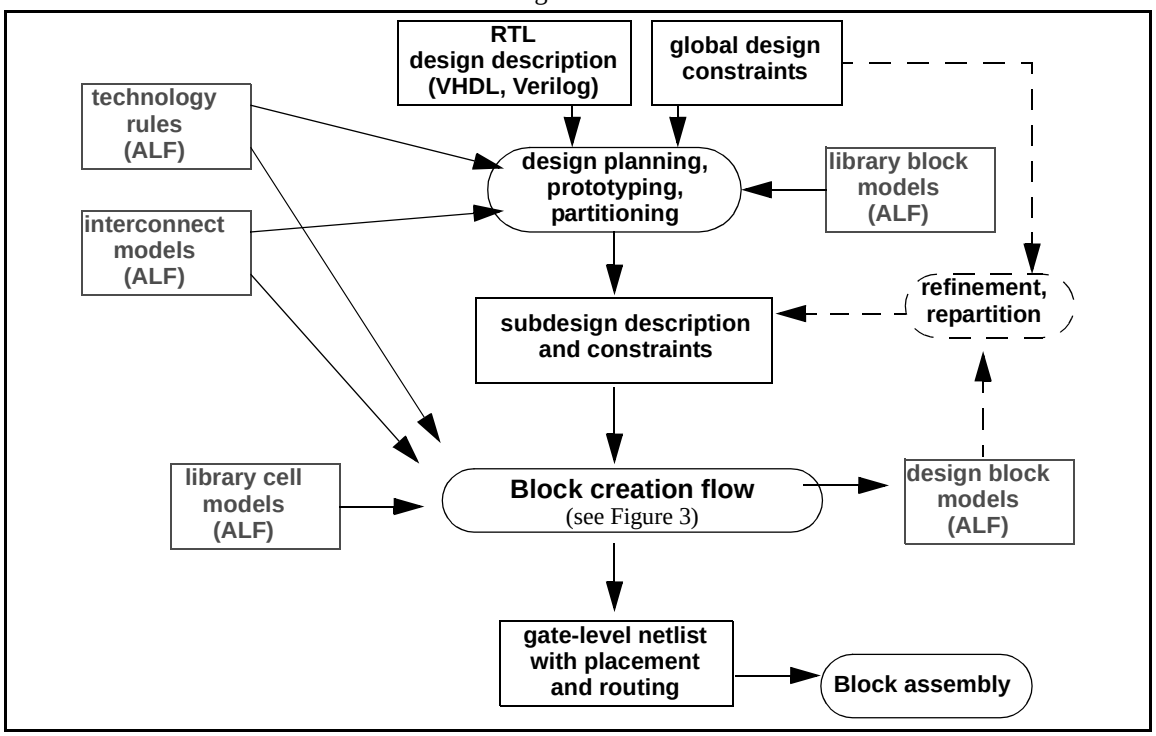

Figure 4—IC prototyping and hierarchical implementation flow

The design planning and prototyping application uses predefined models of blocks as library elements, referred to as "library block models." The design is partitioned into subdesigns. The block creation flow (see Figure 3), i.e., a combination of block implementation and block characterization, is applied to each subdesign. The applicable library elements for each block are cells. The outputs of the block creation flow are the characterized models of the subdesigns, referred to as "design block models." The design block models can be used to iterate on the design planning application, resulting in a possible refinement and repartitioning of the design. Once the evaluation of each block against the subdesign constraints and the

evaluation of the virtually assembled blocks against the global design constraints are satisfactory, the block implementation results, i.e., the netlist with placement and routing for each block, can actually be assembled to form the IC.

The design of an IC can use a combination of cells, hard blocks and soft blocks, blocks with fixed specification, and parameterizeable blocks as library elements. Some of the library elements are available independent of the design, others are created during and only for the purpose of that particular design. An abstract model for a soft block can be used in conjunction with a more detailed model for a hard block. The abstract model can be replaced with a more detailed model during implementation of the block. Technology rules and interconnect models are used throughout the flow.

In summary, the ALF standard provides a common modeling language for library elements, technology rules, and interconnect models. ALF models at different levels of abstraction can be used concurrently by EDA applications for planning, prototyping, implementation, analysis, optimization, and verification of complex ICs.

1.3 Conventions used in this standard

The syntax for description of lexical and syntax rules uses the following conventions:

```
::=     definition of a syntax rule
|       alternative definition
[item]  an optional item
[item1 | item2 | ... ]
        optional item with alternatives
{item}  optional item that can be repeated
{item1 | item2 | ... }
        optional items with alternatives which can be repeated
item    boldface specifies verbatim usage of a string of characters.
ITEM    uppercase boldface specifies verbatim usage of a keyword.
prefix_item
        prefix in italic is for explanation purpose only
PREFIX_item
        prefix in uppercase italic indicates that a keyword is used
```

NOTE—These conventions do not prescribe usage of uppercase or lowercase characters, as ALF is case-insensitive.

1.4 Contents of this standard

The organization of the remainder of this standard is

- Clause 2 (References) provides references to other applicable standards that are assumed or required for this standard.
- Clause 3 (Definitions) defines terms used throughout the different specifications contained in this standard.
- Clause 4 (Acronyms) defines the acronyms used in this standard.
- Clause 5 (ALF language construction principles) defines the language construction principles used in this standard.
- Clause 6 (Lexical rules) specifies the lexical rules.
- Clause 7 (Generic objects and related statements) defines syntax and semantics of generic objects used in this standard.
- Clause 8 (Library-specific objects and related statements) defines syntax and semantics of library-specific objects used in this standard.

— Clause 9 (Description of functional and physical implementation) defines syntax and semantics of statements related to functional and physical implementation of library elements used in this standard
— Clause 10 (Description of electrical and physical measurements) defines syntax and semantics of statements describing electrical and physical measurements related to library elements used in this standard.
— Annex A, Annex B, Annex C, and Annex D. Following Clause 10 are a series of informative annexes.

2. References

This standard shall be used in conjunction with the following publications. When the following standards are superseded by an approved revision, the revision shall apply.

IEC/IEEE 61691-1-1:2004, Behavioural languages - Part 1-1: Language reference manual.[3,4]

IEC/IEEE 61691-4:2004, Behavioural languages - Part 4: Verilog hardware description language.

IEC/IEEE 61523-3:2004, Delay and power calculation standards - Part 3: Standard Delay Format (SDF) for the electronic design process.

IEEE/ASTM SI 10-2002, American National Standard for Use of the International System of Units (SI): The Modern Metric System.

ISO/IEC 8859-1:1998(E), Information technology—8-bit single-byte coded graphic character sets—Part 1: Latin Alphabet No.1.[5]

ISO/IEC 9899:1999, Programming languages—C.

ISO/IEC 14882:2003, Programming language—C++.

[3]This standard is derived from IEEE 1076-2002 which is a trademark owned by the Institute of Electrical and Electronics Engineers, Incorporated.

[4]IEC/IEEE publications are also available from the Institute of Electrical and Electronics Engineers, 445 Hoes Lane, P.O. Box 1331, Piscataway, NJ 08855-1331, USA (http://standards.ieee.org/).

[5]ISO/IEC publications are available from the ISO Central Secretariat, Case Postale 56, 1 rue de Varembé, CH-1211, Genève 20, Switzerland/Suisse (http://www.iso.ch/). ISO/IEC publications are also available in the United States from Global Engineering Documents, 15 Inverness Way East, Englewood, Colorado 80112, USA (http://global.ihs.com/). Electronic copies are available in the United States from the American National Standards Institute, 25 West 43rd Street, 4th Floor, New York, NY 10036, USA (http://www.ansi.org/).