TECHNICAL REPORT

IEC TR 62017-1

First edition
2001-04

Documentation on design automation subjects –

Part 1:
EDA Industry standards roadmap

Documentation sur les sujets d'automatisation de la conception –

Partie 1:
EDA Industry standards roadmap

Reference number
IEC/TR 62017-1:2001(E)

Publication numbering

As from 1 January 1997 all IEC publications are issued with a designation in the 60000 series. For example, IEC 34-1 is now referred to as IEC 60034-1.

Consolidated editions

The IEC is now publishing consolidated versions of its publications. For example, edition numbers 1.0, 1.1 and 1.2 refer, respectively, to the base publication, the base publication incorporating amendment 1 and the base publication incorporating amendments 1 and 2.

Further information on IEC publications

The technical content of IEC publications is kept under constant review by the IEC, thus ensuring that the content reflects current technology. Information relating to this publication, including its validity, is available in the IEC Catalogue of publications (see below) in addition to new editions, amendments and corrigenda. Information on the subjects under consideration and work in progress undertaken by the technical committee which has prepared this publication, as well as the list of publications issued, is also available from the following:

- **IEC Web Site (www.iec.ch)**

- **Catalogue of IEC publications**

  The on-line catalogue on the IEC web site (www.iec.ch/catlg-e.htm) enables you to search by a variety of criteria including text searches, technical committees and date of publication. On-line information is also available on recently issued publications, withdrawn and replaced publications, as well as corrigenda.

- **IEC Just Published**

  This summary of recently issued publications (www.iec.ch/JP.htm) is also available by email. Please contact the Customer Service Centre (see below) for further information.

- **Customer Service Centre**

  If you have any questions regarding this publication or need further assistance, please contact the Customer Service Centre:

  Email: custserv@iec.ch
  Tel: +41 22 919 02 11
  Fax: +41 22 919 03 00

TECHNICAL REPORT

IEC TR 62017-1

First edition
2001-04

Documentation on design automation subjects –

Part 1:
EDA Industry standards roadmap

Documentation sur les sujets d'automatisation de la conception –

Partie 1:
EDA Industry standards roadmap

International Electrotechnical Commission 3, rue de Varembé Geneva, Switzerland
Telefax: +41 22 919 0300 e-mail: inmail@iec.ch IEC web site http://www.iec.ch

Commission Electrotechnique Internationale
International Electrotechnical Commission
Международная Электротехническая Комиссия

PRICE CODE XE

For price, see current catalogue

Table of contents

FOREWORD **9**

How This Book is Organized **10**

How to Find the Information You Want **10**

1 Introduction **11**

1.1 Charter **11**

1.1.1 Identify the EDA Industry Requirements on EDA Systems 11
1.1.2 Review Status and Current Plans of Related Standards 11
1.1.3 Determine How to Coexist and Migrate to Improved Standards 11
1.1.4 Identify Standards Areas Requiring Improvement 11
1.1.5 Develop a Roadmap to Future Standards 12
1.1.6 Deliver Recommendations and Roadmap Contributions 12
1.1.7 EDA Standards Industry Council 12

1.2 Background **13**

1.2.1 Productivity Improvement 14
1.2.2 Complexity Management 14
1.2.3 Advanced Technology Development 14
1.2.4 Technology Development Funding 15

1.3 Scope **15**

1.3.1 Electronic Design and Test Standards Focus 15
1.3.2 Technology Packages 16
1.3.3 Design Phases 16
1.3.4 Key Electronic Design and Test Interfaces 16

2 Executive Summary **17**

2.1 Roadmaps **17**

2.1.1 Introduction to Roadmap 17
2.1.2 Overview of Design and Test Categories 18
2.1.3 Design System Roadmaps 18
2.1.4 Design Information Roadmaps 21
2.1.5 Key Electronic Design and Test Interface Roadmaps 23

2.2 Recommendations **23**

2.2.1 Key Roadmap Recommendations 23
2.2.2 Coexistence and Migration 27
2.2.3 Areas of Convergence 27
2.2.4 Areas of Acceleration of Work 29
2.2.5 Areas Where New Standards Work is Required 29
2.2.6 Areas Where Additional Roadmap Work is Required 30

2.3 The Standards Development Process 30

2.3.1 Current Standards Development Environment 30
2.3.2 Standards Development Process Recommendations 31

3 Electronic Design and Test Environment 35

3.1 Emerging Paradigm Shifts 35

3.1.1 Innovation in Systems Level Design (Architecture and High level Design Phases) 35
3.1.2 Innovation in Design Process Management 35
3.1.3 Increased Codesign Across Design Disciplines 35
3.1.4 New Architectural and Integration Concepts 35
3.1.5 Changing Business Practices 36
3.1.6 Pay-Per-View for Design Tools 36

3.2 Pressures on Designers and CAD Integrators 36

3.2.1 Exploit Multiple EDA Operating Environments 36
3.2.2 Use Diverse Databases and Formats 36
3.2.3 Use Tools from Multiple Tool Vendors 37
3.2.4 Enforce Design Methodologies and Process Management 37
3.2.5 Reduce (or Maintain) Cycle Time 37
3.2.6 Reduce Design Costs 37
3.2.7 Maximize Return-on-Investment (Price/Performance) 37

4 The Design System (Infrastructure and Tools) 39

4.1 Computing Environment and User Interface 39

4.1.1 Current Environment 39
4.1.2 Requirements 40
4.1.3 Recommendations 41
4.1.4 Roadmap - Computing Environment and User Interface 414

4.2 Design Tool Communication 42

4.2.1 Current Environment 42
4.2.2 Requirements 43
4.2.3 Recommendations 43
4.2.4 Roadmap - Tool Communication 43

4.3 EDA System Extension Language 44

4.3.1 Current Environment 44

4.3.2 Requirements 44
4.3.3 Recommendations 45
4.3.4 Roadmap - EDA System Extension Language 46

4.4 EDA Standards-Based Software Development Environment.....46

4.4.1 Current Environment 46
4.4.2 Requirements 47
4.4.3 Recommendations 47
4.4.4 Roadmap - EDA System Basic Software Development Environment 47

4.5 Intellectual Property Protection for Design Data Objects.....50

4.5.1 Current Environment 50
4.5.2 Requirements 50
4.5.3 Recommendations 51
4.5.4 Roadmap - Standards for Design Data Object Asset Protection 51

4.6 Design Management 51

4.6.1 Current Environment 51
4.6.2 Requirements 52
4.6.3 Recommendations 53
4.6.4 Roadmap - Design Management 54

4.7 Design Data Management.....54

4.7.1 Current Environment 55
4.7.2 Requirements 56
4.7.3 Recommendations 57
4.7.4 Roadmap - Design Data Management 57

4.8 Design Process Metrics Management.....57

4.8.1 Current Environment 58
4.8.2 Requirements 58
4.8.3 Recommendations 58
4.8.4 Roadmap - Design Process Metrics Management 58

4.9 Design Tool Management 59

4.9.1 Current Environment 59
4.9.2 Requirements 60
4.9.3 Recommendations 61
4.9.4 Roadmap - Design Tool Management 62

4.10 Resource Management 63

4.10.1 Current Environment 63
4.10.2 Requirements 63
4.10.3 Recommendations 64

4.10.4 Roadmap - Design Resource Management 64
5 The Design Information (Design Data Representation) 65
5.1 Common Topics Across All Design Information 65
5.1.1 Incremental Processing 65
5.1.2 Hierarchical Processing 67
5.1.3 Design Object Naming 68
5.2 Common Topics Across All Design Steps 69
5.2.1 Timing Information 69
5.2.2 Simulation and Test Control 73
5.3 System Level Design 75
5.3.1 Current Environment 75
5.3.2 Requirements 75
5.3.3 Recommendations 77
5.3.4 Roadmap - System Level Design 78
5.4 Detailed Design 79
5.4.1 Current Environment 79
5.4.2 Requirements 80
5.4.3 Recommendations 86
5.4.4 Roadmap - Detailed Design 86
5.5 Design and Technology Re-use 88
5.5.1 Environment 88
5.5.2 Requirements 89
5.5.3 Recommendations 92
5.5.4 Roadmap - Design and Technology Re-Use 92
6 Key Electronic Design and Test Interface.....96
6.1 Manufacturing Build Interface.....96
6.1.1 Current Environment 96
6.1.2 Requirements 96
6.1.3 Recommendations. 97
6.1.4 Roadmap - Manufacturing Build Interface 97
6.2 Manufacturing Test Interface.....98
6.2.1 Current Environment 98
6.2.2 Requirements 98
6.2.3 Recommendations 99
6.2.4 Roadmap - Manufacturing Test Interface 99
6.3 Mechanical Design Interface.....99

6.3.1 Current Environment 99
6.3.2 Requirements 99
6.3.3 Recommendations 100
6.3.4 Roadmap - Mechanical Design Interface 100

6.4 Software Design Interface.....100

6.4.1 Current Environment 100
6.4.2 Requirements 101
6.4.3 Recommendations 102
6.4.4 Roadmap - Software Design Interface 102

Appendix A - The Roadmap Working Groups 103

List of Tables

Table 2.1: Areas of Recommended Standards Convergence 28
Table 2.2: Areas of Recommended Standards Acceleration 29
Table 2.3: Areas of Recommended New Standards Work 29
Table 2.4: Areas of Recommended Additional Roadmap Development 30
Table 5.1: Impact of Design Size on Design Processing Times 81
Table A.1: EDA System Interoperability and Integration Working Group (EII) 105
Table A.2: Design and Data Management Working Group (DDM) 106
Table A.3: Technology Libraries and Models Working Group (TLM) 107

List of Figures

Figure 2.1— Design System Roadmap 20
Figure 2.2— Design Information Roadmap 22
Figure 2.3— Key Design and Test Interfaces Roadmap 23
Figure 2.4— The EDA Industry Standards Roadmap 25
Figure 2.5— Vision of Standards Development 33
Figure 4.1— Open EDA Enterprise Architecture 49
Figure 4.2— Open EDA Data Interoperability Architecture 50

INTERNATIONAL ELECTROTECHNICAL COMMISSION

_____

DOCUMENTATION ON DESIGN AUTOMATION SUBJECTS – Part 1: EDA Industry Standards Roadmap

FOREWORD

1) The IEC (International Electrotechnical Commission) is a worldwide organization for standardization comprising all national electrotechnical committees (IEC National Committees). The object of the 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, the IEC publishes International Standards. 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. The 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 the 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 National Committees.
3) The documents produced have the form of recommendations for international use and are published in the form of standards, technical specifications, technical reports or guides and they are accepted by the National Committees in that sense.
4) In order to promote international unification, IEC National Committees undertake to apply IEC International Standards transparently to the maximum extent possible in their national and regional standards. Any divergence between the IEC Standard and the corresponding national or regional standard shall be clearly indicated in the latter.
5) The IEC provides no marking procedure to indicate its approval and cannot be rendered responsible for any equipment declared to be in conformity with one of its standards.
6) Attention is drawn to the possibility that some of the elements of this technical report may be the subject of patent rights. The IEC shall not be held responsible for identifying any or all such patent rights.

The main task of IEC technical committees is to prepare International Standards. However, a technical committee may propose the publication of a technical report when it has collected data of a different kind from that which is normally published as an International Standard, for example "state of the art".

IEC 62017-1, which is a technical report, has been prepared by IEC technical committee 93: Design automation.

It is based on the *EDA Industry Standards Roadmap - 1996* developed jointly by Silicon Integration Initiative Inc. (formerly CAD framework initiative, Inc.), EDAC and SEMATECH.

The text of this technical report is based on the following documents:

Enquiry draft	Report on voting
93/97/CDV	93/113/RVC

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

This publication has not been drafted in accordance with the ISO/IEC Directives, Part 3.

This document which is purely informative is not to be regarded as an International Standard.

The committee has decided that the contents of this publication will remain unchanged until 2004. At this date, the publication will be

- reconfirmed;
- withdrawn;
- replaced by a revised edition, or
- amended.

How this Book is Organized

This book contains six chapters, summarized below:

- Chapter 1, “Introduction,” describes the charter, the three working groups that contributed to the roadmap, and the scope of the work is detailed.
- Chapter 2, “Executive Summary,” summarizes the key messages to the industry council regarding the requirements and the status of EDA standards with respect to those requirements, the recommendations for standards convergence, acceleration, and new standards work, and the recommended standards roadmaps for each category of requirements. The Executive Summary is designed to summarize the information in the chapters that follow that expand upon the Electronic Design and Test area, discussing the environment, the requirements, and the status, recommendations, and roadmaps in more detail.
- Chapter 3, “Electronic Design and Test Environment,” reviews environmental topics that are relevant to the effective design of complex electronic systems. Key emerging paradigm shifts in the EDA industry and many of the pressures on design and CAD integrator teams are identified and discussed.
- Chapter 4, “The Design System (Infrastructure and Tools),” includes the computing environment and user interface, design tool communication, extension language, software development environment, and the design and data management areas. These topics are primarily domain-independent topics, but with an EDA focus.
- Chapter 5, “The Design Information (Design Data Representation),” addresses the key standards related to design data representations for all key design activities, including system level design (architectural and high level design activities) and detailed design (both logical and physical design in a given technology), as well as preparation for manufacturing build and test.
- Chapter 6, “Key Electronic Design and Test Interfaces,” discusses the interfaces to other design (or co-design) disciplines, and the key standards that relate to those interfaces. Manufacturing build and test Interfaces, software interfaces, and mechanical design interfaces are discussed.

How to Find the Information You Want

Below are some quick access tips to help you find the information you want to read about in this book.

- To access the *key roadmap recommendations*, their priority and timeframe, read Section 2 "Executive Summary".
- To access a given topic’s detailed information including the *environment, requirements, recommendations and roadmap tasks* with descriptions, find the topic in the table of contents and go to the referenced page.
- Those topics that relate to *Design and Data Management* are located in Chapter Four, since they are related to the general design system environment.
- Topics related to *Technology Libraries and Models* are located in Chapter Five, as they are specific representations of EDA design data.

DOCUMENTATION ON DESIGN AUTOMATION SUBJECTS -

Part 1: EDA Industry Standards Roadmap

1 Introduction

1.1 Charter

The EDA Standards Roadmap Workshop was sponsored by the CAD Framework Initiative (CFI), Electronic Design Automation Companies (EDAC), and SEMATECH with participation by interested industry groups. The Workshop was specifically aimed at developing an industry-wide roadmap for development of design and test standards within Electronic Design Automation (EDA).

There were three working groups, each charged with identifying requirements, understanding the current standards environment, and developing a roadmap to improved standards across the next decade for their area of focus:

- EDA Interoperability and Integration Working Group (EII)
- Technology Libraries and Models Working Group (TLM)
- Design and Data Management Working Group (DDM).

The charter of these working groups is discussed below.

1.1.1 Identify the EDA Industry Requirements on EDA Systems

A primary goal was to identify the target requirements of the EDA industry on EDA Systems over the following timeframes:

- Immediate (Short) Term (1995-1998)
- Near Term (1999-2001)
- Long Term (2002-2004 and Beyond).

1.1.2 Review Status and Current Plans of Related Standards

With an understanding of the EDA industry requirements, develop a mapping to the relevant standards involved in supporting those requirements, and review the status and plans of current EDA standards that relate to those requirements.

1.1.3 Determine How to Coexist and Migrate to Improved Standards

Identify the standards changes necessary to meet the requirements, as well as the appropriate coexistence and migration plans from today’s current situation to the target EDA system standards structure.

1.1.4 Identify Standards Areas Requiring Improvement

Identify potential standards convergence opportunities, areas where new focus on existing

standards work is needed, and areas where acceleration of planned standards work is required.

Clearly identify elements of the standards roadmap in the following categories:

- Convergence of Standards in Areas of Overlap
- New Focus on New Work
- Areas Requiring Acceleration.

1.1.5 Develop a Roadmap to Future Standards

Keeping in mind that the perspective of the roadmap should be a global one, develop and deliver a roadmap of key action plans to support the requirements, and itemize the standards-related work tasks required over time (using the definition of timeframe above). The roadmap should include task descriptions of each of the work items in the roadmap.

The task descriptions have different levels of precision depending on the timeframe for implementation. Those tasks required in the Immediate or Short Term (i.e., end of 1998 to coincide with next generation of CMOS.25um) will have more detail than tasks in later timeframes.

The roadmap for the short term (immediate) timeframe should define the steps that the standards organizations must take to achieve the recommended convergent state of standards to support the Technology Roadmaps. The roadmap steps will define the detailed implementation plan for items in the short term. Roadmap items that are targeted for the near or long term may be less defined.

1.1.6 Deliver Recommendations and Roadmap Contributions

CFI, EDAC and SEMATECH will assist the work groups and provide the lead for the final integration of the individual workgroup recommendations and contributions into an overall EDA Standards Roadmap. Tasks will include:

- Prioritize recommended actions
- Document the needed timeframe for implementation
- Seek approval of the Roadmap and its recommendations by the EDA Industry Council.

1.1.7 EDA Standards Industry Council

The Industry Council is comprised of individuals with the credentials and influence to support the roadmap and promote industry adoption. The membership represents a broad range of geographic, academic, and government interests and reflects the major constituencies of the EDA industry. Industry Council members include:

- Joseph Borel, SGS Thomson Microelectronics
- Ron Collett, Collett International
- Joseph Costello, Cadence Design Systems, Inc.
- John Darringer, IBM

- Aart deGeus, Synopsys, Inc.
- William Evans, AT&T
- Richard Goering, EE Times
- Andrew Graham, CAD Framework Initiative, Inc.
- Alain Hanover, EDAC and Viewlogic Systems, Inc.
- Randy Harr, ARPA
- Lambert van den Hoven, Philips Semiconductor
- Lance Mills, Hewlett-Packard
- L.J. Reed, Motorola
- Wally Rhines, Mentor Graphics, Inc.
- Robert Rozeboom, Texas Instruments, Incorporated
- Greg Ledenbach, SEMATECH
- Gadi Singer, Intel Corporation
- Gary Smith, Dataquest
- Kinya Tabuchi, Mitsubishi
- Hitoshi Yoshizawa, NEC.

1.2 Background

The initial input to the working groups included several documents. The requirements described in chapters 4, 5, and 6 represent a summarization of the key requirements identified in these documents and other sources.The key documents include:

- SIA National Technology Roadmap for Semiconductors[1]
- SRC White Paper[2]
- IPC OEM Requirements[3]

As indicated in the SIA National Technology Roadmap for Semiconductors (aka, the SIA Roadmap), the U.S. semiconductor community faces new challenges as it moves towards design and manufacturing of chip feature sizes less than .50 microns. This is not unique to the U.S. and is in fact a global problem. A few of these challenges span the entire spectrum of technology including chip cores, chips, MCMs, and boards (PCAs/PCBs), and they require major industry initiatives to develop effective solutions. The SIA Roadmap states that the magnitude of the challenges listed below demands the special attention of the semiconductor industry leadership.

1. The National Technology Roadmap for Semiconductors, 1994
2. Design Needs for the 21st Century: White Paper, Edited by Dr. James Freedman, VP of Research Integration, Semiconductor Research Corporation, 9/94
3. OEM Requirement as Input to National Technology Roadmap for Electronic Interconnection, 3/14-16/95

1.2.1 Productivity Improvement

The SIA Roadmap suggests that the semiconductor industry will require productivity gains greater than the 30% historical per-year, per-function cost reduction. Achieving projected densities and projected growth will require unprecedented industry cooperation and standardization through consensus. Many standards are required to achieve cost-effective factories; but the EDA standards are key to the success of future design teams struggling to achieve the productivity and density objectives.

1.2.2 Complexity Management

The years 2007-2010 will see maximum chip sizes increase to 350-800M million transistors for microprocessor designs, and 210M-430M gates[1] for ASIC designs. Successful development of designs this size will:

- require very large design teams and enormous effort,
- involve considerable design complexity, and
- result in a huge amount of design data to manage.

Maintaining control of these designs requires innovative design approaches and tools that support hierarchical and incremental design changes. EDA systems need to provide sophisticated design process and information management capabilities well beyond today's. To support the rapid evolution of design environments, new and innovative EDA solutions must be inserted into complex design processes. Widespread use of design re-use technology and complex intellectual property for data and tools, within and between companies, will be a major enabling technology for designs of such complexity.

EDA systems must be put together in ways that allow "technology insertion" of new EDA innovations as they become available. EDA standards that support an evolving design environment are imperative.

1.2.3 Advanced Technology Development

Funding for high cost and long-term research efforts has been reduced both in the U.S. and the rest of the world. The resulting gap in infrastructure must be filled by members of the semiconductor R&D community.

Improvements in software engineering are required; software applications are now fundamental to design, manufacturing, and business processes. Software development is the least perfected of engineering disciplines.

Increasingly, design and manufacturing of complex electronic systems requires a cultural change, from local optimization, to global optimization of technology solutions across multiple engineering disciplines. Changing industrial cultures is a formidable and time-consuming task.

---

1. The National Technology Roadmap for Semiconductiors, Table 2, p. 16, SIA, 1994

1.2.4 Technology Development Funding

Meeting the challenges of the SIA Roadmap will require an increased expenditure of resources on research and development of technology from the already heavy levels of today. The key challenge is to clearly define requirements and find funding strategies that cover all critical needs.

Items 1.2.3 "Advanced Technology Development" and 1.2.4 "Technology Development Funding" above are not primarily within the scope of this work; however, 1.2.1 "Productivity Improvement" and 1.2.2 "Complexity Management" are clearly contained within the scope of work for this EDA Industry Standards for Design and Test Roadmap.

1.3 Scope

This section defines the scope of the EDA Industry Standards Roadmap for the SIA Electronic Design and Test Areas. It includes strategic direction for key electronic design and test standards, with specific focus on the design system infrastructure, tools and design data, and key packaging technologies, across all key design phases.

The scope is focused on EDA, and is specifically focused on:

- EDA *systems and software* standardization (NOT standardization of electronics hardware)
- EDA and key *interfaces* to the EDA domain (NOT standardization of other CAD domains)
- EDA *Roadmap* for standards development (NOT actual development of the standards)
- EDA *standards* roadmap (NOT a design process or algorithm development roadmap)

1.3.1 Electronic Design and Test Standards Focus

Electronic design and test standards focus includes: infrastructure and tools, design and data management, design data representation, and technology libraries and models.

1.3.1.1 The Design System

This section summarizes the standards for the infrastructure surrounding and supporting Electronic Design and Test systems and tools. It also covers all design and data management standards relating to the definition, execution, and management of the electronic design process.

- Infrastructure and Tools

  This includes all standards dealing with the infrastructure surrounding and supporting Electronic Design and Test systems and tools. The subject addresses platform operating systems, communications environment, user interfaces, tool encapsulation, inter-tool communication, and general EDA development environment. These standards promote innovation of new processes and methods for electronic design and test, that can be easily inserted into the design environment (i.e., plug and play). Such standards operate at various levels to bind tools together into design flows, and enable cost-effective multi-vendor flows without requiring tight integration of tools.

- Design and Data Management

  This area includes all design and data management standards areas relating to the definition, execution, and management of the electronic design process and it's associated workflows and data across the enterprise. EDA design tool integration and general tool management standards are also discussed in this area.

1.3.1.2 The Design Information

This section addresses Design and Test design information (data) and the definition and re-use of design technology libraries and models.

- Design Data Representation

  This area includes all standards areas relating to the definition and representation of Electronic Design and Test design information (data) created and used by EDA design tools and/or the design team in the execution of the design process. Examples include such items as netlists, libraries, test benches, and many kinds of in-process data. The purpose of this area is to identify standards for EDA and point tool suppliers in order to support high quality information exchange and data sharing throughout the design process and across a geographically dispersed enterprise.

- Technology Libraries and Models

  Included in this area are all standards relating to the definition and re-use of design technology libraries and models. Key library standards or standards requirements are identified, and focus areas and a roadmap for the next decade is described.

1.3.2 Technology Packages

The following technology and package types are addressed in this roadmap:

- Technology Cores, Cells, and Macrocells (i.e., subsets of chips)
- Chips
- MCMs
- Boards (PCAs, PCBs, backplanes, subassemblies of various types).

1.3.3 Design Phases

The key design phases involved in design and test processes will be addressed, including:

- System Level Design (i.e., Architectural and High Level Design), and
- Detailed Design (i.e., Detailed Logic Design and Detailed Physical Design).

1.3.4 Key Electronic Design and Test Interfaces

The key interfaces involved between electronic design and test and other related disciplines will be addressed, including:

- Manufacturing Build Interface
- Manufacturing Test Interface
- Mechanical Design Interface
- Software Design Interface.

2 Executive Summary

This chapter of the document summarizes the findings of the working groups, and provides the essential information from which the Industry Council review presentations were drawn. Highlights and key messages to the industry council are included for:

- recommended standards roadmaps for each of the key areas,
- recommendations for standards convergence, acceleration, and new areas for development,
- recommendations for a modernized standards development process.

2.1 Roadmaps

The working groups and the charter are briefly reviewed below, along with an overview of the categories of roadmaps developed, and then the roadmaps for each category.

2.1.1 Introduction to Roadmap

The roadmap is designed to be a high level plan for the Electronic Design Automation (EDA) industry to converge on a common set of standards for the next decade. A summary of the charter and scope of this work is summarized below.

The EDA Standards Roadmap Workshop was sponsored by the CAD Framework Initiative (CFI), Electronic Design Automation Companies (EDAC), and SEMATECH with participation by interested industry groups. The Workshop was specifically aimed at developing an industry-wide roadmap for development of design and test standards within EDA.

There were three working groups, each charged with identifying requirements, understanding the current standards environment, and developing a roadmap to improved standards over the next decade in their area of focus:

- EDA Interoperability and Integration Working Group (EII)
- Technology Libraries and Models Working Group (TLM)
- Design and Data Management Working Group (DDM).

The charter of these working groups was as follows:

- Identify the EDA Industry Requirements on EDA Systems over time

  - Immediate (Short) Term (1995-1998)
  - Near Term (1999-2001)
  - Long Term (2002-2004 and Beyond).

- Review Status and Current Plans of Related Standards
- Determine How to Coexist and Migrate to Improved Standards
- Identify Standards Areas Requiring Improvement

- Convergence of Standards in Areas of Overlap
- New Focus on New Work
- Areas Requiring Acceleration.

• Develop a Roadmap to the Future Standards.

2.1.2 Overview of Design and Test Categories

The information in this document and in this executive summary section is organized into categories as follows:

• Design System (Infrastructure and Tools)

This part of the executive summary highlights the key roadmap items for standards related to the

- Computing Environment and User Interface,
- Design Tool Communication,
- EDA System Extension Language,
- EDA Standards-Based Software Development Environment, and
- Design and Data management areas.

• Design Information (Design Data Representations)

This part of the executive summary highlights the key roadmap items for standards in the following design information areas:

- Common topics across all design information (such as incremental processing, hierarchical processing, and design object naming),
- Common topics across all design steps (such as timing, simulation controls),
- System Level Design (i.e. architectural and high level design standards)
- Detailed Design (i.e., detailed logical and physical design standards)
- Design and Technology Re-Use topics, including Technology Libraries and Models.

• Key Design and Test Interfaces

In this section, we summarize the key roadmap items relative to the key interfaces associated with the general area of “design and test”. The interfaces to manufacturing build and test are discussed, as well as high level roadmaps for mechanical design and software/ hardware codesign.

2.1.3 Design System Roadmaps

This section summarizes the specific roadmaps for each of the categories. Details on each of the roadmap items can be found in chapters 4-6. Note that each roadmap item as shown in the figures is numbered according to the roadmap chapter and section. In the figures, the following scheme is used to indicate priority

• High priority items are indicated by the darkest fill color (or red)

• Medium priority items are indicated by a lighter fill color (or blue)

• Low priority items are indicated by the lightest fill color (or white).

Note: *All* items are considered important or they would not be in the roadmap.

The start and stop points for items in the roadmaps indicate a mapping of the consensus of the workgroups as to the priority of the requirements (high, medium, low) and the timeframe for the item (i.e., short term, near term, long term).

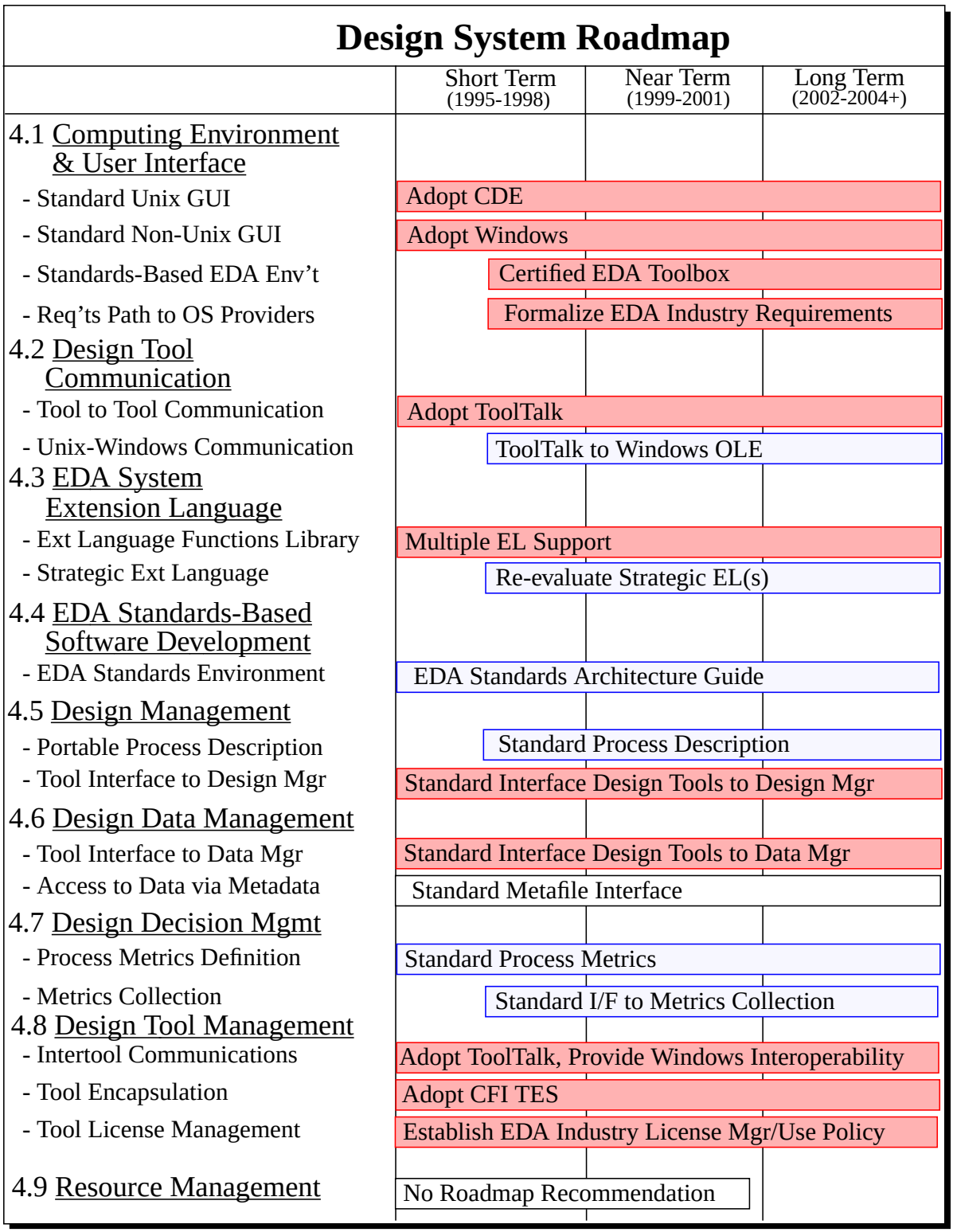

Figure 2.1—Design System Roadmap

2.1.4 Design Information Roadmaps

In this section, the roadmaps that pertain to the standards for design data representation are addressed. The roadmap shown in Figure 2.2— "Design Information Roadmap" includes standards for the following areas:

- Common Topics Across All Design Information, which include:
  - Incremental Processing
  - Hierarchical Processing
  - Design Object Naming
- Common Topics Across All Design Steps, which include:
  - Timing Information
  - Simulation/Test Control
- System Level Design (i.e. architectural and high level design standards)
- Detailed Design (i.e., detailed logical and physical design standards)
- Design and Technology Re-Use topics, including Technology Libraries and Models.

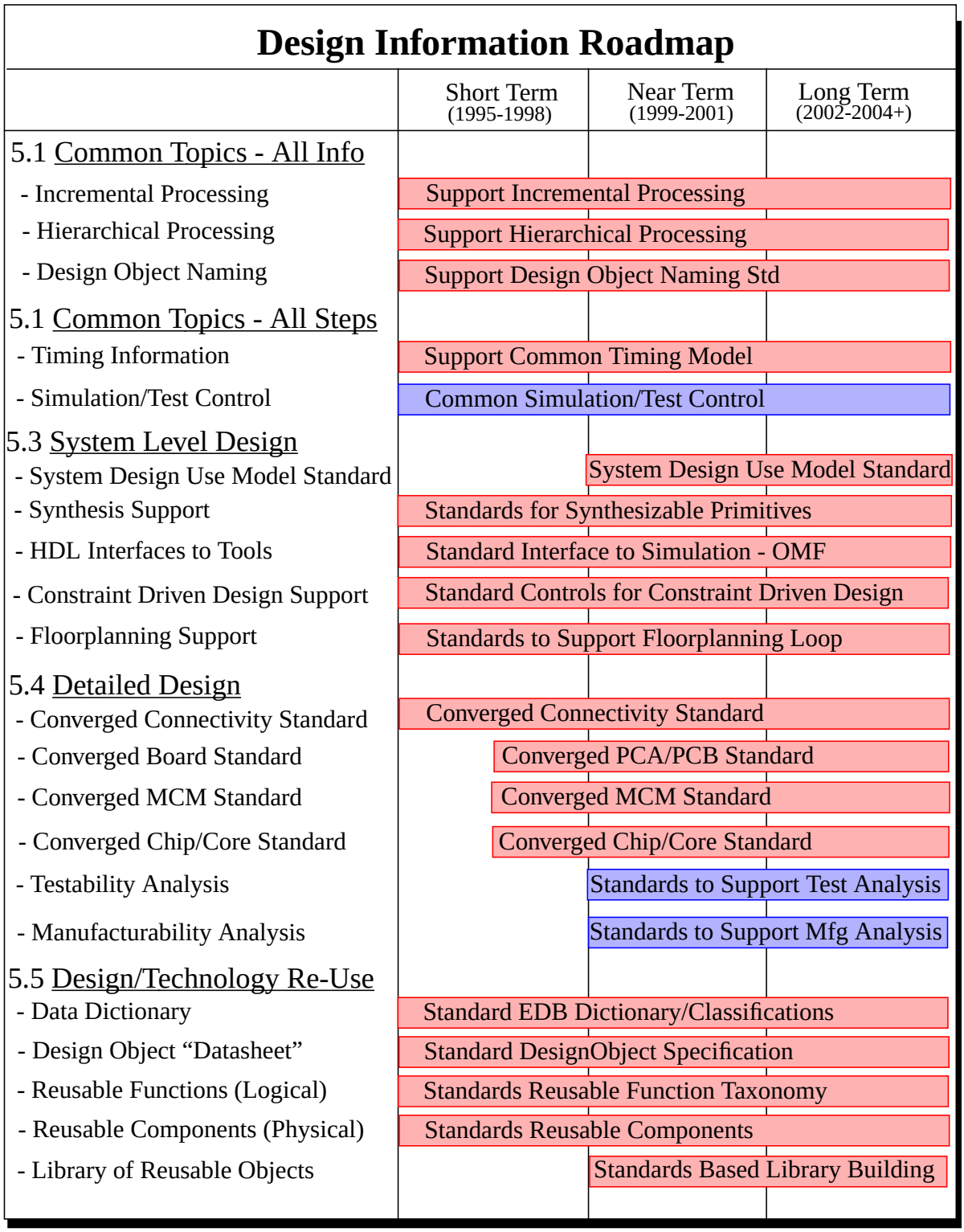

Figure 2.2—Design Information Roadmap

2.1.5 Key Electronic Design and Test Interface Roadmaps

Key Electronic Design and Test Interface Roadmaps shown in Figure 2.3— "Key Design and Test Interfaces Roadmap" include:

- Manufacturing Build Interface
- Manufacturing Test Interface
- Mechanical Design Interface
- Software Design Interface.

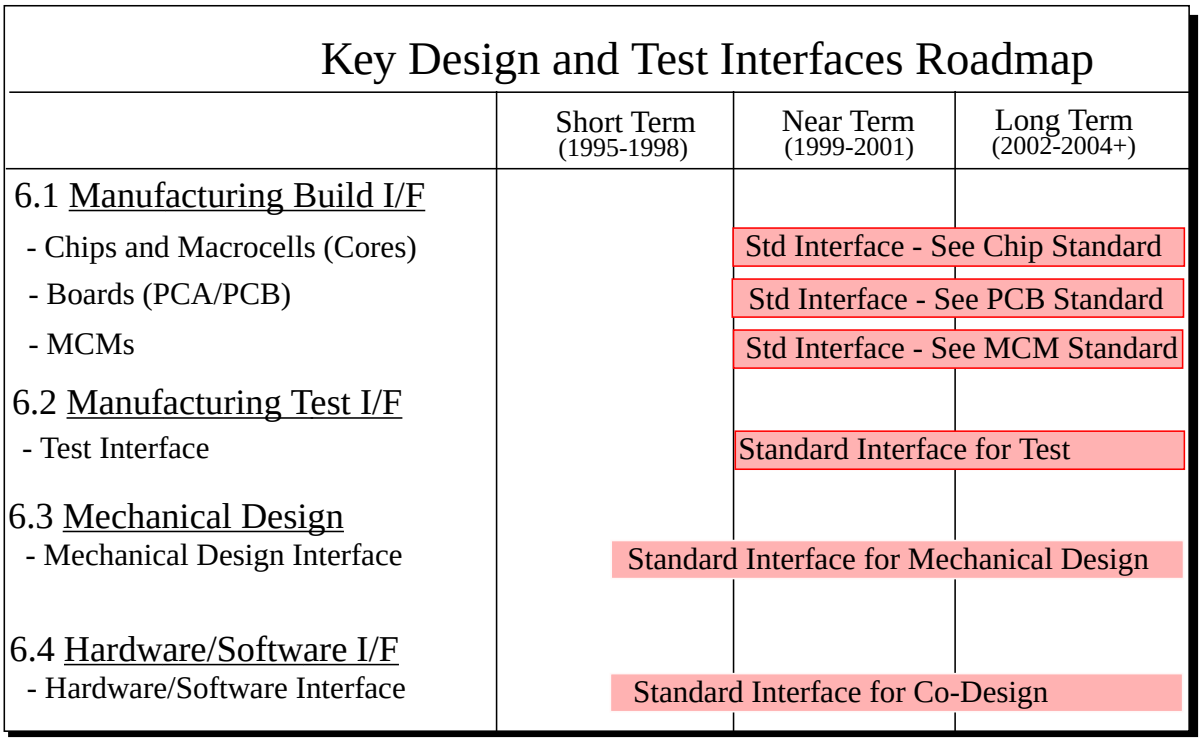

Figure 2.3—Key Design and Test Interfaces Roadmap

2.2 Recommendations

In this section, the key recommendations are summarized. Additional information which provides backup and support for these recommendations is detailed in chapters 4-6 of this book.

2.2.1 Key Roadmap Recommendations

The key recommendations of the EDA Industry Standards Roadmap are highlighted in Figure 2.4— "The EDA Industry Standards Roadmap". Along the roadmap on the chart are the timeframes; short term (through 1998), near term (through 2001), and long term (2004 and beyond). The column entitled “Current Standards” lists key current standards in use by the

EDA industry today, arranged in groups entitled “Systems Design” and “Detailed Design.” Along the bottom of the roadmap, we see “Co-Existence”, “Migration”, and “Evolution” which characterize some of the key goals of the roadmap strategy. As we move along the roadmap from left to right, there are a series of recommended EDA Industry Standards:

- Common Connectivity Standard

  In this initial release, which is based on the common core information model from the EDIF - CFI DR converged model work, the base is created from which *all* the other design data representation standards developments evolve. This release establishes, for the first time, a file format (in this case an interchange language format, based on EDIF 3 0 0), and a programming interface (based on CFI DR 1.X) which are based upon the *same* information model. In addition, note the “Hierarchical, Incremental, Naming” at the base of the diagram; this is denoting the significant importance of providing support for the hierarchical and incremental processing of design data in *all* future standards developments.

  From this point forward, the idea is to add functionality to the base connectivity standard for various “use models,” based on such design topics as timing or package type (e.g., PCB or chip), or even system design models (based on VHDL or Verilog use models), including the necessary support for hierarchical and incremental processing, and at all times have a matched pair of file format and programming interface (PI). Section 2.3.2.1 "Technical Approach" discusses this approach to standards development in more detail. Refer to 5.4.4.1 "Converged Industry Standard for Logical Connectivity" for additional information on this specific roadmap item.

- PCBs and MCM Standards

  In this phase, the physical information required to support board level packages including PCBs (printed circuit boards) and MCMs (multi-chip modules) is added to the common information model for connectivity developed in the previous step. There are two standards (or use models) released in this phase; one for boards, and one for MCMs. A file format and PI are both available and are based on the extensions made to the common information model as described above. This work should begin as soon as possible, and the physical information should be initially based on the EDIF 3 5 0 for PCB (and eventually EDIF 4 0 0 for MCM) information models.

- Chips and Cores/Macrocells Standard

  Similarly, in this phase, use model standards for physical design data for chips and subsets of chips (cores or macrocells) are supported. A file format and PI are both available and are based on the extensions made to the common information model.

  Over time, the defacto standards of today which contain chip/core information will converge into this industry wide standard. For example, the roadmap says that the timing information which is today contained in SDF files, could in the future be contained in a “chip/core” standards-compliant applications database. Because of the standards strategy of information modeling, followed by a file format (or language) and a PI, it is possible to co-exist with legacy standards (e.g., the SDF), while migrating to a new standard (the chip/core standards). It is imperative that vendor tools be developed as soon as possible to this new standard to meet the design requirments detailed in Chapter 5; however, because

of the ability to co-exist with legacy standards while migrating towards new standards, new tools which operate from the new standard can exchange information with legacy tools. EDA vendors can migrate to the new standard when their business situation dictates. At any point in time, the industry would have some vendor tools which operate on legacy standards, and other tools which capitalize on the new standard, and the strategy recognizes the reality of this, and supports it.

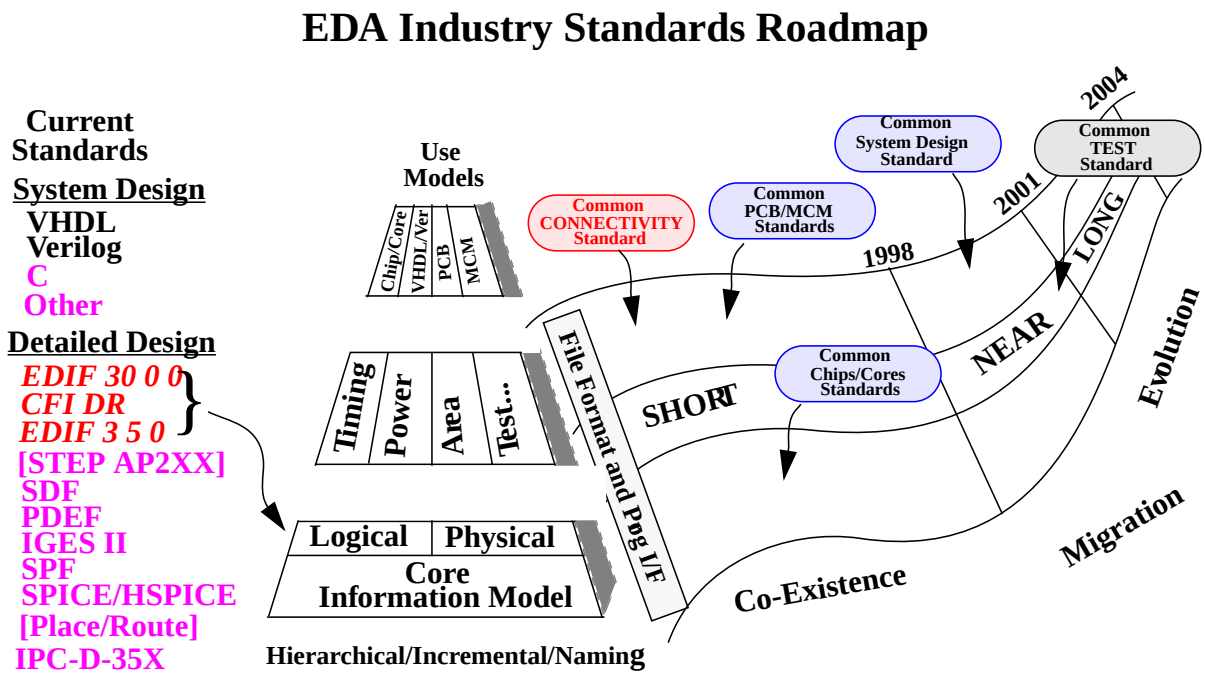

Figure 2.4—The EDA Industry Standards Roadmap

- System Design Standards

  In this phase, which is envisioned later in this decade, such as in the near term, a common information model and *standard for system level design* is developed and added to the information model base from the previous step. Because VHDL is significantly more comprehensive than Verilog for system design, it is recommended that we start to build the initial system design standard base using VHDL as a base from which to determine the information model. Similarly, the information model developed from the above step can be extended if necessary to completely cover the information model requirements for Verilog. Additional requirements to support any additional new system design language requirements can also be factored into the information model as well in a similar fashion. From this converged information model, a system design standard with a PI can be developed, with appropriate mappings to any HDL (e.g., VHDL and Verilog).

  Using this strategy, for VHDL as an example, would allow the vision to be realized for a new VHDL use model standard; i.e., there is an information model, from which a file for-

mat (i.e., the language, which in this example is VHDL and already defined) and a programming interface is developed. A new VHDL use model standard can be released in the near term which includes both the language and a PI to access that information in PI-compliant tools. Over the long term, the strategy states that for the file format, the system design standard should move to a common exchange file format which supports incremental processing.

It is important to note that to the extent that the information modeling efforts between VHDL and Verilog overlap (i.e., the information being modeled is the same information), then the PI to access that information is *identical*. Over time, as extensions are made to the system design information model (e.g., for analog support), the PI for those extensions is the same for both VHDL-based and Verilog-based customers, even though the file format representations (i.e., the languages) are different. It is anticipated that over time, more and more applications will migrate to the PI approach for data access in cases where data sharing is desired. It would be *very desirable* that the information modeling efforts described above be converged between VHDL and Verilog, but it is not required to achieve VHDL-based or Verilog-based use model standards which have a PI as well as a language.

It should also be noted that this strategy for standards development supports the identification of areas where VHDL and Verilog have a direct mapping (e.g., the areas where the information model and PI are the same), and also those areas where the information models are different, and hence where their use model standards would be different. The creation of language translators would also be facilitated by this information modeling work; however, it must also be noted that because of the differences in VHDL and Verilog, translation between those languages is definitely not automatic, and human intervention may well be required in such translation. Of course, the goal is really to not require data translation, but to use the information modeling based strategy to help us co-exist with the existing legacy languages while we migrate towards an information and data sharing approach.

Defining an information model derived file format (designed for effective tool to tool data exchange of incremental design information, not designed to be human readable) would make it possible for any *specific* language dependency to be reduced over time, and this strategy enables new HDL languages, including graphical representations (non-textual) to be directly supported by the system design standard and PI, and thus be less dependent on language form. Refer to Figure 4.2— "Open EDA Data Interoperability Architecture" for additional comments on co-existing with legacy languages and file formats. Ultimately, it will be a combination of the users and the tool developers who will determine the rate of movement from the traditional system design language based approach to more picture-based system design approaches which are to be supported by the system design standards. Again, this strategy for standards evolution is independent of that rate of change.

- Common Test Standard

  In the area of test, it is also envisioned that later in the decade, perhaps in the near term, a common test information model for test information should be developed, from which existing legacy test standards can be supported, and yet a common test standard can be developed to enable new design and test support software and hardware to be developed to

meet emerging design and test requirements. In the same fashion as in the other standards, the test standard also would have a file format and PI available based on the common test information model.

2.2.2 Coexistence and Migration

The roadmap recommendations described above would be extremely difficult to achieve without an effective method to coexist with today's standards while we are attempting to migrate to a better standards-based future.

In order to support coexistence and migration, it must be possible to support legacy standards such as SDF, PDEF, and other file formats, while the industry migrates to a more focused strategic standard such as a common PI and related file format(s)/languages. To support co-existence and migration, a family of translators from each important legacy language (file format) to information model compliant design data repositories must be developed and made available to the industry at large. With this approach, there would be a need for only *one* certified translator for each language in each direction (i.e., one for import, and one for export).

This strategy also makes it possible to gracefully coexist with and migrate to new standards-based design data repositories which include *mixtures* of legacy data and data based upon the new standards roadmap. For example, tools which depend upon SDF files today can still operate without change, and new PI-based design data repositories can import/export SDF transparently through the PI (i.e., the PI knows where and how the data is to be stored or retrieved). Tools which are being rewritten (e.g., to support incremental or hierarchical design) can access that data directly via a PI, or via a common standards-based file format.

To the extent that this standards development process (i.e., information modeling, file format and PI) approach is adopted by industry, it would enable legacy design languages and interchange file formats to eventually phase out as primary design representations. Coexistence with legacy formats would be supported by the translator set, and new tool development would be done totally to a new file format and PI based on the information model. The PI approach also insulates the tool developers from the specific technology used to actually store and retrieve the information, such as relational data base technology, object oriented technology, (e.g., OMG CORBA), and so on. Client applications are also completely insulated from the specifics of the database implementation, and may be distributed across a LAN or WAN, through the use of the PI approach. As a result of this approach, innovation is enabled, yet tools can be implemented in an open EDA environment.

2.2.3 Areas of Convergence

This section summarizes areas where ongoing standards work has considerable overlap. This overlap has been identified, and the recommendations contained here are that certain standards be strategically converged. Table 2.1: "Areas of Recommended Standards Convergence" summarizes the recommended convergence of standards.

As indicated in Figure 2.4— "The EDA Industry Standards Roadmap", examples where standards efforts must be converged include:

- The convergence of EDIF 3 0 0 and CFI DR 1.X into an industry standard which encom-

passes both of them. In the figure, the base core information model and the common connectivity standard reflect this approach.

- The convergence of common design topics such as timing and physical parameters including parasitics and floorplanning, and placement and wiring information must be converged over time into the common information model. Over time, this would imply the *gradual phaseout* of several current standards, including:

  - SDF
  - PDEF
  - IGESII
  - SPF
  - potentially SPICE/HSPICE
  - various floorplanning, and place and wire file formats.

  Note that the phaseout of the above standards was described as gradual; this is because the strategy allows this to be achieved over time, based upon EDA vendor development capabilities and user demand, with the “coexist and migrate” strategy described above. The proposed strategy enables all EDA tools to be based on the file (or PI) based interface to design data, as long as they have value to users; but the goal is to get to the PI.

- The convergence of standards for all types of packages and levels of design based on a common information model from which various “use models” for each package type are developed. Use models for boards (PCA/PCB), MCMs, one for chips/cores/macrocells, and a system design language are recommended.

- The convergence of standards related to the manufacturing test interface is also possible (but less studied and understood) and is based on the same strategy as for the design and build standards for packages; i.e., determine the information model requirements based on existing or legacy standards, and develop a file format and PI which supports the converged information model. As in the package standards convergence examples above, the test standards in use today should converge with potential phaseout of selected test-related standards, including:

  - STEP AP211
  - WAVES
  - IEEE DASC short term common chip tester interface.

Table 2.1: Areas of Recommended Standards Convergence

Current Standard	Recommended Standards Convergence
EDIF	5.4.4.1 “Converged Industry Standard for Logical Connectivity
CFI DR	5.4.4.1 “Converged Industry Standard for Logical Connectivity
SDF	5.4.4.1 “Converged Industry Standard for Logical Connectivity
PDEF	5.4.4.1 “Converged Industry Standard for Logical Connectivity
IGES II	5.4.4.1 “Converged Industry Standard for Logical Connectivity

Table 2.1: Areas of Recommended Standards Convergence

Current Standard	Recommended Standards Convergence
SPF	5.4.4.1 “Converged Industry Standard for Logical Connectivity
SPICE/HSPICE	5.4.4.1 “Converged Industry Standard for Logical Connectivity

2.2.4 Areas of Acceleration of Work

In this section are areas of ongoing standards work which need an accelerated rate of development to meet current and future design requirements. A boost in funding and/or resources or EDA vendor focus may be required to accelerate the work of development or adoption of a standard. Refer to Table 2.2: "Areas of Recommended Standards Acceleration" for a list of areas where standards work should be accelerated.

Table 2.2: Areas of Recommended Standards Acceleration

Current Work	Recommended Standards Acceleration
OMF	5.3.4.3 “Standard Interfaces to Design Analysis Tools”
ToolTalk	4.9.4.1 “Intertool Communications: Adopt ToolTalk...”, others
Tool Management	4.9.4.3 “ Establish EDA Industry License Manager/Use Policy
DCL Project	5.2.1.4.3 “Complete Delay Project and Extend Beyond ASICs”

Examples in this category include: development of HDL independent standards (e.g., OMF).

2.2.5 Areas Where New Standards Work is Required

In this section are areas where gaps exist in standards, or there are no standards at all. New standards are required to meet critical design requirements, now and in the future. Refer to Table 2.3: "Areas of Recommended New Standards Work" for a list of areas where new standards work is required.

Table 2.3: Areas of Recommended New Standards Work

Related Standards	Recommended New Standards Work
ToolTalk, OLE 2	4.2.4.2 “Provide Windows Interoperablilty for ToolTalk”
VHDL, Verilog, C	5.3.4.1 “Standards for System Level Design”
Library Standards	5.5.4 “Roadmap - Design and Technology Re-Use”

2.2.6 Areas Where Additional Roadmap Work is Required

In this section, areas which require additional study to complete a detailed roadmap are described. In these areas, the working groups ran out of time and/or resources, and it was determined that additional follow-on work is required before conclusions can be reached. Refer to Table 2.4: "Areas of Recommended Additional Roadmap Development"for a list of areas where additional roadmap development work is required.

Table 2.4: Areas of Recommended Additional Roadmap Development

Related Standards	Recommended Additional Roadmap Development Work
Library Standards	5.5.4 “Roadmap - Design and Technology Re-Use”
Board Standards	5.4.4.2 “Converged Industry Standard for Board Packages”
Data Management	Chapter 4 Design Data, Resource, and Release Management
Manufacturing Test	6.2 Manufacturing Test Interface
Mechanical Interface	6.3 Mechanical Design Interface
Software Interface	6.4 Software Design Interface (Hardware/Software Co-Design)

2.3 The Standards Development Process

This section describes the current standards environment and provides recommendations for the future.

2.3.1 Current Standards Development Environment

The most widely used standards support tool interoperability through the use of “standard” interface file formats. Standards efforts working on the definition of these file formats tend to be somewhat disjointed, and to a degree, competitive. This leads to the necessity for industry players to either support all of these different standards, or pick the one(s) they feel most likely to satisfy their customers. To complicate matters, the standards tend to differ not only in format, but also in content and in the basic structure of their information models. For example, the basic information model for EDIF 3 0 0 electrical connectivity is different from that of CFI DR 1.X or STEP AP2XX. This leads to a further need for translation software to convert between these different standards, which is costly, short lived, and prone to errors.

To meet the challenges of the future, there are a number of standards related needs that must get into focus:

- EDA standards efforts must be harmonized in line with a single “roadmap” (as described in this document). Some harmonization efforts have started, such as between CFI and EDIF, but this is “ad hoc” and without long term targets or goals. There is considerable confusion and frustration across the EDA industry, the ASIC suppliers, and the end-user design community. A roadmap-driven standards effort will offer a strong and stabilizing base from

which to build more effective long term set of standards.

- The time period from standardization to the appearance in commercial products is much too long. A considerable reduction in this time should be a mandatory requirement in reforming the standards development process. The time required for standard identification, definition, industry acceptance, and certifiably accurate implementation needs to be addressed, and requires much closer cooperation between EDA industry participants.

2.3.2 Standards Development Process Recommendations

This section describes the technical vision of standards development and proposes a management model for standardizaton of future developments.

2.3.2.1 Technical Approach

This section describes the vision and the idealized model of standards development from a technical viewpoint, and the strategy for coexistence and migration towards the ideal model from today's situation.

In general, for a given EDA standards area that needs significant work (and where it makes sense), the conceptual process of building a standard should include:

- Definition of an information model (in EXPRESS)
- A programming interface (built from the information model)

  A software programming interface (PI) is designed to support an access method for data *sharing* and direct data access (i.e., no data exchange required). This approach can also be used for high performance data *transfer* between PI-compliant tools. When implemented as a programming interface appropriately, this approach is inherently hierarchical and incremental in nature. This approach can also be used to provide a strategic PI interface between client and provider applications which enables the support of legacy file formats while coexisting and migrating to more strategic forms of design data and PI access. This means that a client application can request design data through a PI interface, and be relatively independent from the actual source of the data (e.g., an SDF or a new converged standard which contains the timing information).

- A file format (built from the same information model)

  The file format approach is designed to support the *exchange/archival* of design information where that makes sense; e.g., when passing data between two companies, or a development site and a manufacturing site (where the programming interface concept of data sharing may not apply). Clearly, file formats also provide a base from which to *translate* design data where translation is an effective mechanism.

  It is possible that a binary computer readable (not human readable) file format which is based on the information model can be developed which is suitable for data archival and incremental data exchange between information model compliant applications.

Both the file format and the programming interface should be developed and *delivered simultaneously,* as new releases of the standard evolve.

To better convey this concept, refer to Figure 2.5— "Vision of Standards Development". The concept is as follows:

- The Common Connectivity Model

  A common core information model is developed (in EXPRESS) for fundamental design connectivity (which is the same connectivity model for all package types). That common connectivity model must also ensure and record the mapping and relationship of all logical and physical information about the entities being connected.

- The Design Topics Model

  In this layer, the idea of various design topics such as timing, power, area, and test information get modeled (also in EXPRESS). Each additional design topic for which a standard is required get added in this fashion. As an example, in the illustration, a timing value for a net is shown as .5ps; this timing information would be “annotated” to the base connectivity model as information regarding the net timing information between a specific output net of one block and a specific input pin of another block.

- The “Use Model” Standards

  In addition to the information modeled in the previous layers, there is certain additional information required to support “use models” for specific types of packages and the release to manufacturing of design information for those package types. For example, there is information that is unique to board (PCA/PCB) packages, MCMs, and also for chips/cores/Macrocells. The example in the diagram is one of a PCB. Therefore, in general, there is a subset of the information in the common connectivity model, the design topics layer, and the physical package type, that when considered collectively make up the use model for that type of package.

  Similarly, there could also be a use model for system design (e.g., a VHDL use model).

  Any test of compliance to a “use model” standard actually implies compliance to proper processing of the information per the use model involved.

The concept here is one where the design data information model evolves from a common core information model, upon which specific design topics can extend that core model (e.g., timing), and from which various use models (e.g. a package type such as “chip”) can be further developed. Similarly, a use model for system level design can be developed. From each of these use models a “standard” can be developed.

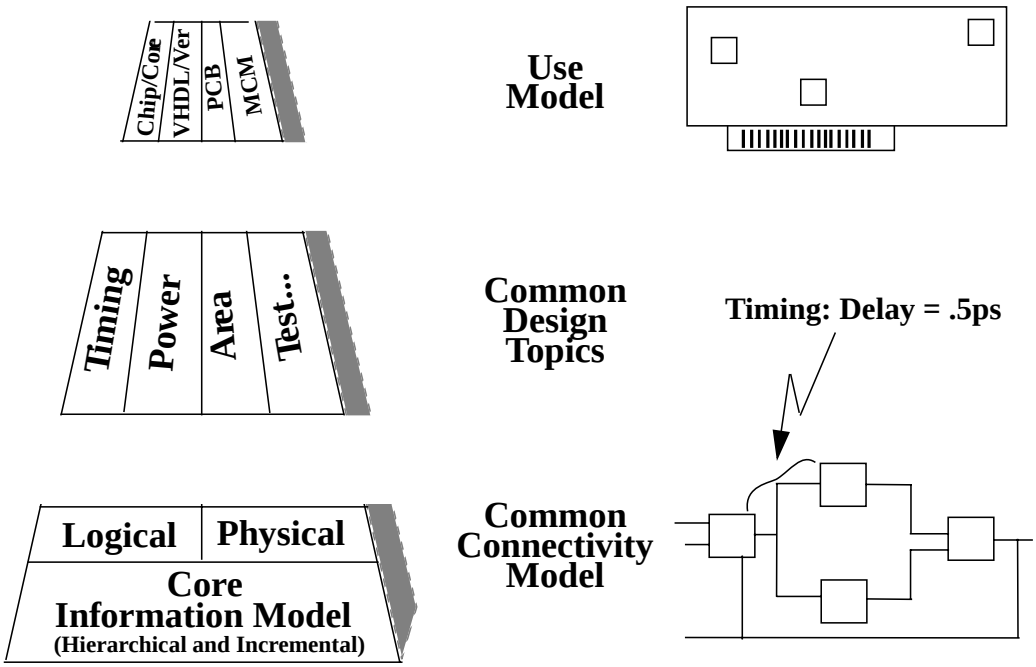

Figure 2.5—Vision of Standards Development

2.3.2.2 Standards Management Model

This section overviews the existing standards situation, and proposes a management strategy supporting the cooperative and focused development and adoption of future standards.

It is easy to observe that there are many standards organizations with standards that they develop and promote. It is also clear that there is confusion in the EDA marketplace, and unnecessary overlap and competition in selected standards areas. This problem must be addressed by the industry council.

It is recommended that the industry council come to a common agreement that *all* design data representation standards developed for the EDA industry be developed under the process described in this document, and when ready (i.e., accepted by industry), be submitted to a *single* standards body (e.g., IEEE, EIA, etc.) for formal standardization process and life cycle management.

2.3.2.3 Pilot Programs and Prototype Development

Any new standard or significant new release of a standard should have a prototype implementation using the draft level of the standard to form a candidate standard.

Such prototyping or pilot programs should be targeted at real-world design problems using real EDA tools, modified to support the draft standards involved. It is via these pilot programs that key issues with the draft standards get identified and resolved before balloting as a new standard. Coexestence with legacy standards as well as support to migrate to the new standards must be demonstrated.

2.3.2.4 Test Case Development/Management

Test cases for large chip designs should be gathered (or constructed) and made publicly available to facilitate and promote use in the testing of new tools developed by vendors and universities.

This would be helpful in creating tools that whose function can be demonstrated in real-world EDA environments and be more production ready. These test cases must be kept current and meaningful and a plan must be put in place to ensure that the test cases do not become outdated. Such test cases could also be part of a certification process or any important benchmark process for standards. There are at least two potential repositories for these test cases:

- the SRC/SEMATECH supported benchmarking facility at NCSU by Franc Brglez
- CFI, as part of a broad EDA industry “certification laboratory.”

2.3.2.5 Conformance/Certification Plan

Any new standard or significant new release of a standard should have a conformance or certification plan and test suite to promote uniform application of the standard and thus promote true interoperability among tools implementing the standard. Where applicable, certification should be accomplished by and through the use of appropriate test suites to certify the implementation.

2.3.2.6 Productization Support

Industry support of and by the EDA vendors to develop new or upgraded tools to the standards must be provided throughout the above process, from concept through prototyping, through to actual released products which support the standards involved. A strong and concerted effort (including the necessary resources and funding) is required to actually get products in the marketplace which are based on or depend upon new standards.

3 Electronic Design and Test Environment

In this chapter, environmental topics that are relevant to the effective design of complex electronic systems, are discussed. Many of the pressures on design and CAD integrator teams are identified and discussed.

3.1 Emerging Paradigm Shifts

Significant paradigm shifts within the electronic design community are dramatically changing the face of design and practices, and imposing new requirements on EDA tools and systems. These are discussed in this section.

3.1.1 Innovation in Systems Level Design (Architecture and High level Design Phases)

Many of the design process and tool innovations will occur in the very front end of the design process. New tools are emerging to support the specification and analysis of design architectures, including performance evaluations, design trade-off analysis, hardware software co-design, and estimations of various factors such as life-cycle costs.

3.1.2 Innovation in Design Process Management

As the size of chips and electronic systems in general increases, the size of design teams to develop them increases. The complexity of managing design flows and iterative design changes will increase dramatically. Design teams will be dispersed geographically. The designer productivity and increased codesign requirements will drive innovations in the area of concurrent design.

3.1.3 Increased Codesign Across Design Disciplines

Technology evolution, e.g. Deep Submicron semiconductor technology, is forcing major discontinuities in traditional design methods. The complexity and scale of integration, and significant cost of design errors, promotes a reevaluation of design practice and a great increase in “co-design” or collaborative (and concurrent) design, early in the design process and spanning multiple design disciplines. Parallel efforts in disciplines such as design for test, mechanical, software, hardware/software, and electrical design are being driven earlier and earlier into the design process.

3.1.4 New Architectural and Integration Concepts

The massive size of chips is enabling the integration and increasing re-use of chip cores and other design objects coming from different application domains (e.g. telecommunications, computing, biomedical) into one design. This will demand whole new architectural concepts and may include much more programmability to turn general architectures into application-

specific products. This in turn will drive the development of new EDA tools, design processes and methodologies, and libraries to support these concepts.

3.1.5 Changing Business Practices

The business of developing electronics-based systems is undergoing significant change, and this is expected to continue during the next decade. Design, and EDA environments for design, are now subjected to stringent Return-On-Investment (ROI) analysis. This analysis fuels trends such as outsourcing of selected phases of product development including the actual functional design, physical design, or manufacturing of a design object. EDA is not keeping up with semiconductor process technology. Re-use is a major design consideration. There will be much more manufacturing knowledge (build, test, cost, yield, etc.) brought early into the design process. These and other changing design practices will lead to different business practices such as for intellectual property (e.g., chip cores, design processes and others).

3.1.6 Pay-Per-View for Design Tools

The geographically distributed product development environment necessary in the next decade will also enable other new EDA design tool marketing and distribution approaches where tools are “rented” for use in design projects. This will enable small enterprises or even individuals to be able to afford previously unreachable design tools. For example, designs could be securely shipped to “design centers” for physical design (common today), or tools could be licensed to designers on a pay-per-view basis to do that design activity.

3.2 Pressures on Designers and CAD Integrators

Electronic engineering environments range from small enterprises using loosely connected tools to very large and complex, distributed, heterogeneous virtual enterprises. This range of design environments contends with significant pressures on design and CAD integrator teams as shown below.

3.2.1 Exploit Multiple EDA Operating Environments

In the workstation environment, UNIX is currently the mainstay for EDA tools, with growing interest in Microsoft Windows NT. However, on the PC platform, Microsoft Windows is dominant, with a limited chance of any other significant players in the operating system environment. Today, many companies have design flows that exploit more than one of these operating environments. This is expected to continue. There is a growing urgency for a single design flow to effectively use both operating environments as if they were one environment.

3.2.2 Use Diverse Databases and Formats

Many design groups do not use Product Data Management (PDM) systems today, relying on a loose network of tools fed by various forms of netlist files. As designs get larger and re-use becomes more prevalent, this must change. There are many different PDM systems in use across the EDA industry today. Product data management systems exploit relational databases

as well as object-oriented database technology. This area of database technology is a rapidly emerging technology and this is expected to continue. The challenge for EDA systems is to be able to exploit and capitalize on the best data management technologies as they emerge.

3.2.3 Use Tools from Multiple Tool Vendors

There is a critical need for new EDA tools which help designers meet goals for minimum time to market of their products. Designer productivity is a major issue now, and the pressure on designer productivity will only increase as technology moves into deep submicron. Commercial EDA companies will continue to strive to develop new tools and capabilities that meet productivity needs. It is clear that the “best of class” tools will not all come from one vendor, especially when the time pressure of the SIA Roadmap is considered. Tools from multiple vendors need to behave as if they come from one vendor because CAD integrators and designers will want to use the best practice tool in their design flows, since it helps establish their competitive advantage.

3.2.4 Enforce Design Methodologies and Process Management

As designs become larger and more complex, the team of designers will also increase in size. In order to keep track of the state of various elements of the design (e.g., chip cores), hierarchical and incremental approaches will be required. Managing the state of the design process for each of these design elements, in a concurrent engineering environment, will be a major challenge. It is expected that improvements in process and workflow management will be major issues in the next decade as the complexity of designs increases.

3.2.5 Reduce (or Maintain) Cycle Time

There is a continuing pressure on design teams to minimize the cycle time associated with the development cycle, in the face of dramatically increased complexity of the electronics. While the design cycle time is currently getting longer, there is a strong demand to increase the “circuits per day” productivity metrics even as design complexity grows. This must be accomplished without sacrificing design quality.

3.2.6 Reduce Design Costs

As always, there are pressures to minimize design costs. Major contributors to reduced development costs include reduction of design schedules, and production of a design that is “correct” the first time into manufacturing. Thus, there is constant pressure to shorten cycles and at the same time, maintain or improve quality.

3.2.7 Maximize Return-on-Investment (Price/Performance)

CAD integrators and design groups are well aware that their EDA systems design environment is a costly factor in the total cost of developing a product. At the same time, the product could not possibly be developed without significant investments in design technology. The goal is always to minimize product development time, at maximum price/performance of

physical assets such as workstations, PC's, operating system and communications software, and EDA software from multiple vendors. Measuring cost-of-ownership and return-on-investment for design technology is becoming a common objective in many design groups. There is a growing pressure on design groups to run themselves like a business, with investment in design technology being a significant portion of their operating costs.

These pressures on designers and on the CAD Integrators that support them imply several key requirements on the EDA System, and on the design information contained within the EDA System. The Design System Environment and the associated EDA domain data standards are both key elements of the environment.

This environment creates requirements in major categories that are identified as the Design System Environment, and the Design Information Environment. Additional environmental topics within each category are discussed in the subsequent chapters.

4 The Design System (Infrastructure and Tools)

This chapter includes the computing environment and user interface, design tool communication, extension language, software development environment, and several design and data management areas. These topics are primarily domain-independent topics, but with an EDA focus.

4.1 Computing Environment and User Interface

The computing environment of the next decade will include UNIX and Microsoft Windows operating system environments, and potentially some others. Each of these environments have certain implications as described briefly below.

4.1.1 Current Environment

Two main platforms are in widespread use for EDA applications today; desktop and high-end server engineering workstations running variants of UNIX, and “PC” type desktop machines running variants of Microsoft Windows. The most complex chip designs tend to be done using UNIX applications, while PCs with Windows are used for selected PCB designs, FPGA and Programmable IC designs, and some low-end ASIC design (as applicable for selected portions of the design process).

The most popular platforms for EDA software use today include the UNIX workstation. The favored computing environments include:

- SunSoft SunOS, migrating towards Solaris 2.X and follow-ons
- Hewlett Packard HP-UX
- IBM AIX
- DEC OSF/1.

UNIX running on engineering workstations will continue to be a significant platform in many design groups for the next decade. This is due to many factors, including legacy investments in hardware, EDA software, designer training and familiarity, development of design processes and methods, and continued development of the workstation environment by manufacturers to offer competitive price/performance, particularly for very large computer servers for high-end design applications, such as large custom chip designs. All major vendors are members of the COSE alliance, and are, or will very shortly be, compliant with CDE (Common Desktop Environment), that will be the standard graphical user interface and window management environment.

In the PC world, Microsoft’s Windows (Windows 95, Windows NT, etc.) is a clear leader. Also, OLE 2 and Microsoft endorsed CAD standards continue to lay groundwork for future EDA applications on these platforms. It is anticipated that Microsoft Windows and its variants will be the dominant PC operating environment over the next decade. Some leading analysts

have indicated that Windows will emerge as a more widely-significant EDA operating environment, though a significant investment in EDA applications redevelopment for that environment will be required. The availability of back-porting kits for Windows-based applications to run on UNIX workstations, as well as on PC platforms based on Windows, will continue to promote a merger of the EDA marketplace into one large market divided into overlapping subsets, rather than the more distinct and separate markets of today.

Apple OS 7 is the environment for legacy Apple hardware, with OS 7.5 on current Apple PowerPC platforms. Current announced industry strategies from Apple indicate PowerPC platforms will be migrating towards the “Copland” operating system environment, that is Apple’s answer to Windows 95. There are no major EDA applications on this platform today. IBM’s OS/2 Warp, or Taligent, or any follow-ons are not envisioned as contenders in the EDA marketplace at this time. While none of these operating environments are expected to be major players in the EDA world in the next decade, they should be monitored.

Generally, the direction of the computing environment is towards being cross-platform; i.e., the hardware (workstation or PC) is becoming less relevant. It is anticipated that over the next decade, UNIX will be available on PC platforms and that Windows will be available on workstations.

In addition, given the prediction that future designs will involve groups of geographically separated designers, it is expected that networked computing will become the new paradigm for the design environment in the not-too-distant future, allowing groups of people to communicate and cooperate on a single design. This technology is in its formative stages, with Lotus and Novell as the current leaders, and Microsoft promising its own support based on Windows 95. It is too early to tell at this time who will set the norm for this type of computing environment, but the requirements are relatively clear.

4.1.2 Requirements

Below are listed the key requirements in this area.

4.1.2.1 Consistent User Operating Environment

In the geographically dispersed multiple operating environment described above, the important point is that, regardless of the suppliers, the environment, as seen by the user, must provide the same capabilities and “look-and-feel”, independent of the underlying hardware and software.

Investments in EDA design system platforms, software, environments, methods, and user training need to be optimized as these elements evolve, and are replaced, or updated. Legacy investments will be optimized if the learning curve for designers moving to new platforms or OS environments is minimized. Since the coexistence of UNIX and Windows based platforms and applications is now occurring, there is both a strong need, and real opportunity to identify standards for common operating environments, and for commercial industry to realize and adopt these standards.

In order to maximize the effectiveness of design teams, the operating environment and user interface of the design system and its tools play an important role. Standards need to be established that minimize the learning curve for designers moving to new platforms or new operat-

ing system environments, whether they are UNIX-based or Windows-based. In many ways, the UNIX and Windows environments have migrated to the point that they are similar enough to each other that a user of one environment can quickly adapt to the other. Only recently has the UNIX world had the ability to exploit a common user interface that is similar to that of Windows (and Mac), with the advent of the Common Desktop Environment (CDE). There is a requirement for EDA vendors to adopt these operating environment standards in order for designers to maximize the EDA and design technology resources, regardless of platform and operating environment.

4.1.2.2 Consistent EDA Environment

In addition to a general domain-independent user interface, EDA users, CAD integrators, and EDA vendors would benefit from a consistent vendor-independent EDA environment, that includes “Consistent User Operating Environment” above, and also provides a “standard EDA toolbox” that addresses the requirements listed in the rest of this chapter. These standards-based EDA tools include such things as the EDA Message Dictionary for Tool Communication, Tool Encapsulation support, extension language support, and potentially an EDA software development environment (e.g., for promotion of EDA standards-based tool development in the university environment). Each of these examples of EDA appliances must be “certified compliant” to the standards it supports to gain customer acceptance as a totally compliant EDA user (or developer) environment.

4.1.3 Recommendations

From a computing environment and user interface perspective, the EII working group feels that there is “enough” consistency between the user interfaces of CDE, Windows, and Macintosh that there are no major unresolved issues in this area.

The working group recognizes *both* UNIX and Windows as players in the next decade, and without attempting to explicitly pick a winner or emphasize one over the other, recommends that a “coexistence and migration” strategy be developed and supported between them.

This recommendation should be periodically reviewed as new environments evolve (such as IBM’s Taligent or Apple’s Copland).

The next point of focus should be to provide the EDA user community and possibly the EDA developer community with a standards-compliant environment for EDA software development. Thus, a set of compliant EDA tools should be made readily available to the EDA industry quickly to establish a solid design system computing environment and graphical user interface foundation based upon key EDA standards from which future innovations can evolve.

4.1.4 Roadmap - Computing Environment and User Interface

This section provides recommendations for the graphical user interface and EDA development tools.

4.1.4.1 UNIX: Adopt CDE as Graphical User Interface

The recommendation on this item is to adopt CDE on UNIX platforms as the standard for graphical user interface desktop management. Interoperability with the Windows environment is an emerging need. See "4.2.4.2 "Provide Windows Interoperability with ToolTalk", " for additional information related to tool interoperability between UNIX and Windows applications.

This item should be adopted as high priority immediately (i.e., in the short term).

4.1.4.2 Windows: Adopt Windows Graphical User Interface

The recommendation for this item is to adopt Windows on PC (and/or workstation) platforms where justified. This situation should be monitored over the next decade as Windows environments evolve.

This item should be adopted as high priority immediately (i.e., in the short term).

4.1.4.3 Standard Base Operating Environment for EDA

This roadmap item is to define, develop and deliver a base set of certified EDA tools in a package that meets requirements for domain-independent and selected EDA domain needs in the area of tool communication and encapsulation, EDA system extension language support, and potentially selected EDA domain data translators. This tool suite should also provide UNIX and Windows interoperability in a standard way. The EDA development environment is another candidate suite of tools.

This item should be pursued as medium priority for the short term.

4.1.4.4 Establish Formal Path for EDA Industry Requirements on OS Providers

A specific process and path for identification and management of EDA industry requirements should be developed to provide focus to the OS providers on the key needs of the EDA industry. The recommendation here is to assign a neutral EDA industry representative to the OS providers, and that all OS requirement be funneled to that body.

This item should be adopted as high priority immediately (i.e., in the short term).

4.2 Design Tool Communication

This section documents the major design tool issues and provides recommendations for inter-tool communication standards.

4.2.1 Current Environment

Electronic design systems are rapidly evolving to geographically dispersed, network based environments. Major design tool improvements are needed to meet the challenges of the future. The designers are struggling to manage the design process with the massive amounts of data involved. Approaches are needed that enable tools to perform cooperative computing via collaborative efforts on design problems. It must be possible for tools to interact in intelligent ways via inter-tool communication and messaging, reaching across the worldwide network.

4.2.2 Requirements

4.2.2.1 Standards for Inter-Tool Communications

There is a requirement to reduce design cycle time in key design loops by maximizing inter-tool communication and interoperability performance. Through the use of dynamic inter-tool communication between active applications of incremental design changes, as opposed to sequential file translation and transfer of entire design sections, the overall cycle time for key design process loops could be significantly reduced. The required participation of the design engineer in these design loops can be greatly reduced to improve designer productivity. Standard inter-tool communications technologies are required to support this communication.

In addition, such a technology would enable EDA design tools to also achieve a higher level of independence from other EDA support technology such as Product Data Management (PDM) technology (if messages were used to interface to the database tool being used by a tool or enterprise), and from the environment in general. Tools such as workflow or process managers, or any EDA tool, could then communicate with PDM systems in a fashion that enables the insertion of new technology whenever it becomes available. Related information on the topic of inter-tool communication are in section 4.9 "Design Tool Management", 4.6 "Design Management", and 4.7 "Design Data Management".

4.2.3 Recommendations

The ToolTalk messaging facility has already been endorsed by CFI as the standard inter-tool messaging mechanism. CFI has developed an EDA Message Dictionary (draft) standard that meets all of the above requirements, thus the recommendation for ToolTalk with the appropriate EDA Message Dictionary extensions. This can meet all known requirements in this category in the UNIX environment.

In addition, extensions to ToolTalk must be provided to meet requirements for inter-tool communication between tools that are running in the Windows environment and those in the UNIX environment. Such interoperability would best be provided by the operating environment suppliers (i.e., the UNIX or Windows providers). In the absence of that solution, alternative suppliers of such functionality could potentially be found.

4.2.4 Roadmap - Tool Communication

This section includes the recommendations for UNIX and Windows platforms.

4.2.4.1 UNIX: Adopt ToolTalk as the Standard ITC Mechanism

The recommendation on this item is to adopt ToolTalk on UNIX platforms as the standard for inter-tool communication.

This item should be adopted as high priority immediately (i.e., in the short term).

4.2.4.2 Provide Windows Interoperability with ToolTalk

The recommendation on this item is to enable inter-tool communication between Windows-based software and UNIX applications. The recommendation is that a Windows OLE to ToolTalk interface be developed to enable that communication.

This item is a “new standard”, that needs funding, and it should be pursued with medium priority immediately (i.e., in the short term).

4.3 EDA System Extension Language

This section includes the recommendations for EDA systems and toolsets.

4.3.1 Current Environment

Extension languages are an integral part of most current EDA systems and toolsets. They provide designers and CAD integrators with an important mechanism to “glue” tools, often from multiple sources, together into design workflows and processes.

Popular extension languages in use today including SKILL, AMPLE, and Scheme-based extension languages such as the CFI Extension Language (EL). In addition, scripting languages such as PERL (and TK/Tcl) are being frequently used to extend EDA systems.

The result today is the unavoidably wide variety of proprietary and non-proprietary extension language scripts found in most real EDA system environments. The multiplicity of languages, and the proprietary nature of each, means that major EDA design systems functionality that uses extension language form, is difficult and expensive to maintain, to migrate to other toolsets, and to migrate across releases of the EDA tools and design system. Such code is often treated as “throwaway” code, and as a result, is poorly written and documented. This further reduces its long term effectiveness in providing designer productivity. There is currently no plan to migrate to a more strategic language.

4.3.2 Requirements

This section includes the recommendations for an EDA System Extension Language (EL) and EL Functions Library.

4.3.2.1 Standard EDA System Extension Language (EL) Functions Library

A standard EL Functions Library that provides access to EDA design data and design system objects (via a set of standard reusable EL Functions Library, including access to EDA standards-based facilities for design data representation, inter-tool communication, tool encapsulation, user interface constructs, etc.), is required. This library of reusable software objects offers functions and capabilities that:

- Can be used by the application developer or by the CAD integrator/user
- provides a consistent graphical user interface where needed
- offers a consistent set of application controls to CAD integrators/users as well as application developers
- is accessible from all popular extension languages; today this includes CFI EL (Scheme) and PERL.

4.3.2.2 Standard EDA System Extension Language (EL)

In the short (immediate) term, in order to support legacy extension language contributions, multiple extension languages may need to be supported. However, over the near and long term, a single strategic EDA extension language (EL) standard is ultimately required to enable the creation of portable, reusable, migratable, and well-crafted and documented extensions by designers and CAD integrators. This EL must have the following characteristics:

- Easy to learn and use
- Support interpretive (rapid easy development) and compiled (efficient to use) forms of execution
- Provides access to EDA design data and design system objects (via a set of standard reusable EL Functions Library, including access to EDA standards-based facilities for design data representation, inter-tool communication, tool encapsulation, user interface constructs, etc.) as indicated above
- Allow binding of C/C++ code
- Portable across all vendor tools and operating platforms (and where tools support access to the EL Library, they operate to give identical results)
- Support dynamic module/library loading (to isolate the EL library functionality from the EL itself)
- Other characteristics as defined in 4.1 "Computing Environment and User Interface".

4.3.3 Recommendations

Given the volume of legacy EDA code developed in the above languages, a strategy must be devised to support legacy code while attempting to migrate to a more strategic solution.

The CFI Extension Language (based on Scheme, which is an IEEE standard) has been previously recommended for support by all EDA vendors and tools, as a standard extension language, with limited success. PERL has emerged as perhaps a leading contender for the most popular extension language. Additional study to determine which of these candidates (or others) should be selected as the strategic EL for the EDA industry, is recommended. There is no mandatory reason why *one* EL must be selected in the short and near term.

What is perhaps more important, is that access to EDA data, messaging, and other objects be adequately supported, in a standard way, via an EL Library. This library should contain the functions necessary to support the writing of the necessary glue code independent of the EL used. Additional requirements as they emerge should be added to the EL library rather than leading to the creation of new or extended proprietary extension languages.

In addition, extensions to enable PERL to access the EL Library facilities via an API should be supported, along with tools to assist users with eventual migration to the strategic EL.

EDA vendors should develop translation and migration tools to assist in migration to the standard EL at some future major release. EDA vendors should stop extending their proprietary languages, while continuing to maintain them.

4.3.4 Roadmap - EDA System Extension Language

This section provides recommendations for a standardized EDA system extension language and functions library.

4.3.4.1 Provide Multiple EL Access to EL Functions Library

Appropriate API bindings to support multiple EL language access to a common EL Functions Library need to be developed. Languages for which this access is required include CFI EL and PERL.

The CFI EL Library meets some of the documented requirements for an EDA extension language library, and is recommended as the base for the strategic industry standard EL Library. The library of functions that make up the existing CFI EL Library should be supported and extended, to meet new requirements (e.g., DR object access, TES object access, User Interface access, etc. per 4.3.2.1 "Standard EDA System Extension Language (EL) Functions Library" for EDA Extension Language).

This item should be adopted as high priority immediately (i.e., in the short term).

4.3.4.2 Select and Standardize on Strategic EL Language

Additional work is recommended to build consensus on the best strategic extension language. By focusing on the EL library access method from the short term extension languages (i.e., CFI EL Scheme and PERL), the priority of this task can be moved to later in the short term (i.e., medium priority).

Also, once a strategic EL language has been selected and adopted by the EDA industry, migration tools should be developed by EDA vendors to enable easier migration to the strategic EL when the user chooses to make that move.

4.4 EDA Standards-Based Software Development Environment

This sections provides the requirements and recommendations for the EDA standards based development environment.

4.4.1 Current Environment

The computing environment described in section 2 "Executive Summary", is driving the need for EDA tool developers including EDA vendors, proprietary development in user companies, and in university and other research environments to develop their products to operate on multiple hardware and software environments, including both UNIX and Windows. At the same time, development of applications should be done to a number of various other EDA standards, as documented in this roadmap. Porting or simultaneous development of standards-based tools compatible with all popular environments will be important in ensuring timely availability of tools in customer environments, and in the cost of insertion into commercial use at reduced cost.

Developments in the university and other research environments is now done in an “ad hoc” fashion with no direction from the EDA industry to enable innovative new tool development and easier adoption by the industry.

4.4.2 Requirements

University, research, commercial and in-house developers of EDA tools need standards to assist in reducing the cost of development, and improve the standards-based quality of EDA software solutions for design problems.

There is a requirement for a “standards-based development environment” (i.e., development with a focus on appropriate EDA standards) in which to develop tools that can be released to the multiple customer environments in use today, and in the future. Such a standard development environment would be of major value to developers everywhere.

A good example of the benefits of a standard development environment would be in the University research environment, where innovations produced using the standard development environment would more likely be “adoptable” in the industry; i.e., because the tool was developed to popular EDA standards, and can run on popular EDA operating platforms.

4.4.3 Recommendations

An EDA industry-wide standards architecture document that reflects the recommendations and roadmap steps outlined in this document, should be developed.

4.4.4 Roadmap - EDA System Basic Software Development Environment

4.4.4.1 Open EDA Standards Architecture Guidelines

A task team must be put in place to develop an Open EDA Standards Architecture Guidelines document that defines and documents the guidelines for development of EDA applications to better support “plug and play” in the supported EDA computing environment as prescribed by this roadmap. This document would overview the concepts, architecture, and relevant EDA standards (e.g., computing environments, inter-tool communication, tool encapsulation, design data representation, etc.) which should be used to develop tools that can be easily adopted by industry upon successful completion of the application. This document should become the key reference document for standards-based EDA software development.

An overview of such an architecture is given in Figure 4.1— "Open EDA Enterprise Architecture". There are several important considerations of note in this architecture, including

- The use of inter-tool communications in:
  - tool launching (e.g., by CDE, and by workflow managers, and by frameworks
  - tool to tool communications such as in the interface to data management for such actions as “check-in” or “check-out” of design objects from or to the design workspace
  - tool to tool communications for the handling of incremental design changes as opposed to reprocessing the entire design.
- The use of Tool Encapsulation files (TES - see 4.9 "Design Tool Management") for *all*

EDA system related tools. The TES files enable the automatic encapsulation and integration of tools in a consistent manner independent of the way the tool will be launched (e.g., from CDE, from the workflow manager(s), or from a framework).

- The concept that all tools expect their data to be in the local “design workspace”, and that workflow managers and framework managers should make use of the ITC data management message set to ensure that this is true, so that legacy tools can operate in such environments without change. In addition, tools which would like to capitalize on interfacing directly with the data management services layer *may* do so with the same ITC messages, but they are not required to do so.
- Access to the EDA System Extension Language Functions Library from any of the standard EDA Extension Languages (see 4.3 "EDA System Extension Language") supports access to EDA system objects (i.e., ITC messages, design objects in the design workspace, TES files, graphics support, etc.). This facility makes it easy to extend the system, using a standards-based extension language and functions library.

This item should be adopted as medium priority immediately (i.e., in the short term).

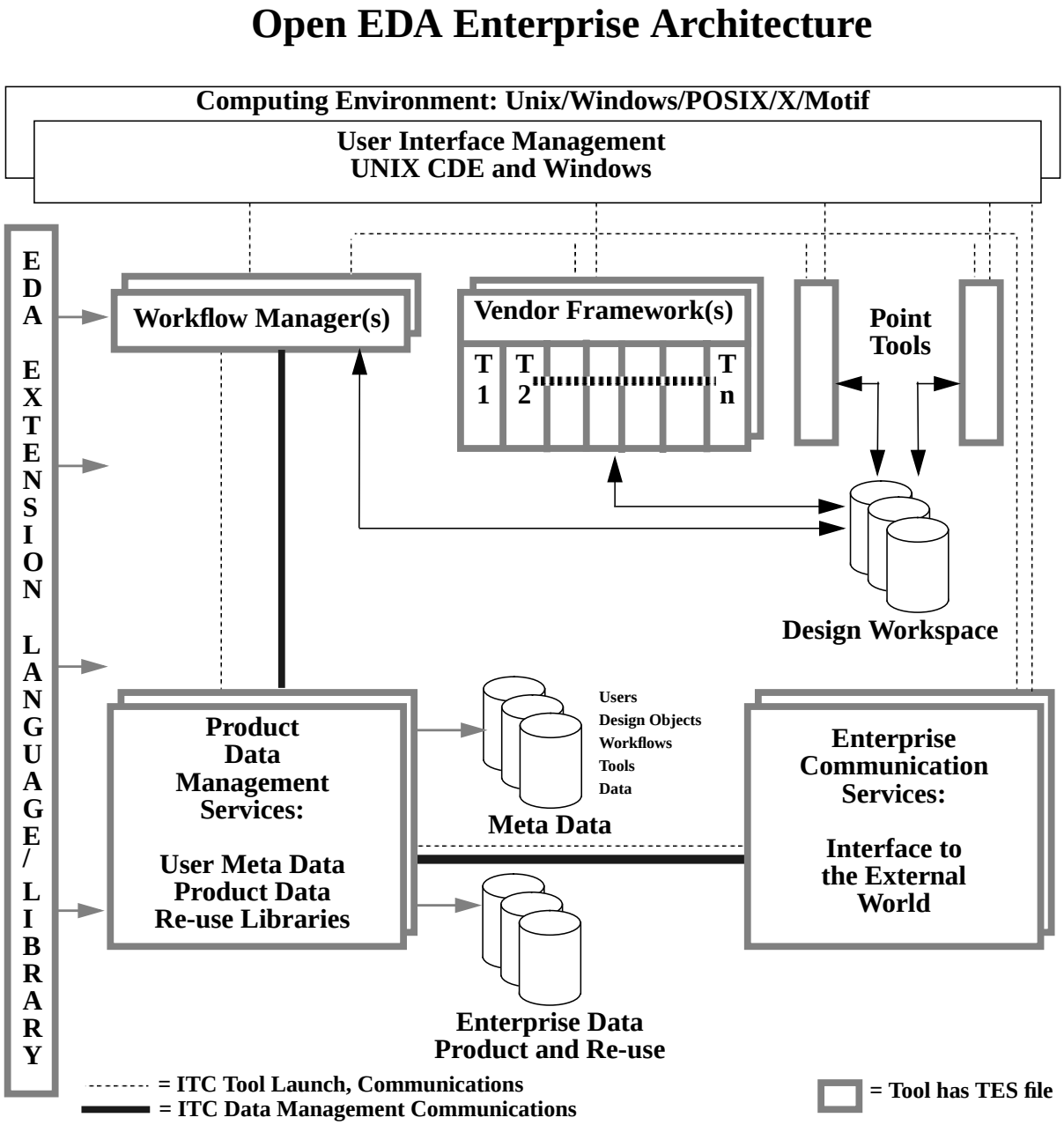

Figure 4.1—Open EDA Enterprise Architecture

Another important part of the Open EDA Standards Architecture is the design data standards, and the data interoperability architecture which supports those standards in a way that co-existence and migration is supported. As indicated in Figure 4.2— "Open EDA Data Interoperability Architecture", a family of information model compliant client applications can be developed for each important legacy language or file format, as well as to vendor proprietary

databases, which can interface to the industry standard (e.g., the chip design representation standard). These client applications exist for both directions (i.e., import and export).

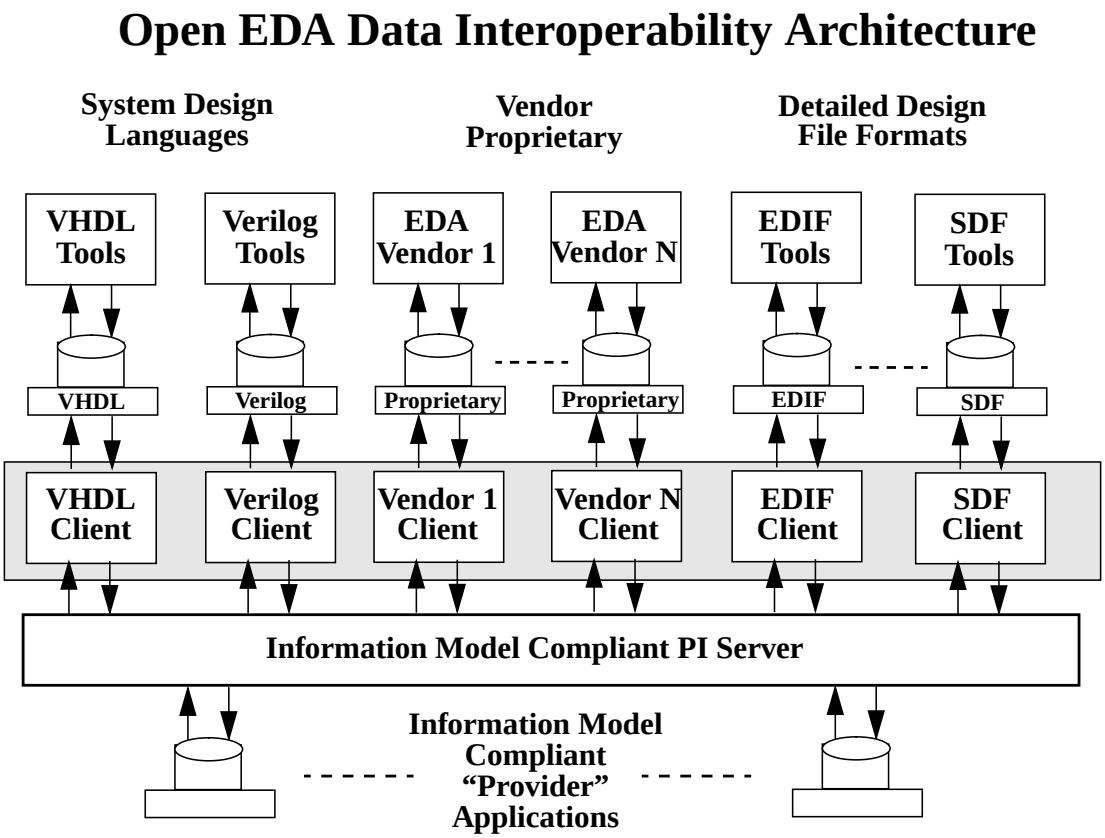

Figure 4.2—Open EDA Data Interoperability Architecture

4.5 Intellectual Property Protection for Design Data Objects

This section addresses intellectual property protection with respect to design data objects. The issue of license management for design tools is discussed in 4.9 "Design Tool Management".

4.5.1 Current Environment

In the case of asset protection and license management for design data objects, there are *no* standards.

4.5.2 Requirements

4.5.2.1 Standards for Design Data Object Asset Protection

There are requirements for standards for license management that support protection of intellectual property assets for design data objects. There is some overlap in the requirements to

protect design data object assets and the requirements to protect design tool assets. The need for asset protection spans a variety of design data objects, including re-use items such as designs for re-use and various technology rules. Each of these examples (and many more) demand an effective solution to protect intellectual property. It must be possible for this information to be electronically distributed over a geographically dispersed network.

4.5.3 Recommendations

While this topic is important to the EDA industry (both developers and users), it is also viewed as a topic that is largely beyond the scope of EDA and more of the scope of the computing environment in general. Solutions to this problem are ultimately expected to come from the OS providers.

4.5.4 Roadmap - Standards for Design Data Object Asset Protection

While this topic is important to the EDA industry, it is believed that the solution must ultimately come from the OS providers. A more strategic solution is anticipated in the short or near term because it is extremely important to commerce on the internet. Therefore, it is recommended that the EDA industry drive to (and wait for) that enterprise-wide platform independent solution from the OS providers.

4.6 Design Management

Design management (or workflow management) encompasses automated support to aid or enforce a prescribed design methodology across a set of design steps and design tools. Elements of design management support include:

- Formal definition of a design methodology (i.e. the design process or workflow)
- Completeness checking of design steps and control or assistance in determining the next allowed process step(s)
- Association of design data types and design tools for each design step
- Auditing of the design process to assure valid design hand-off
- Design status, or state, accounting

Design projects involving large design teams or large numbers of design steps and tools are finding it increasingly necessary to incorporate design management automation to assure consistency and accuracy in the overall design process. While often true that design methodologies are loosely defined and enforced, there is a growing interest in formality due to the increased complexities of semiconductor design, the number of designers involved with a design entity and the necessary interaction across multiple design domains and geographic locations.

4.6.1 Current Environment

Although there is a diversity of design environments within the industry today, the general

norm is a rather loose definition of the overall business process. The EDA design environment tends to be captured within scripts, that define the tools required for the various design tasks, written in extension languages such as SKILL, AMPLE, PERL, Scheme, and TK/Tcl. The overall business process tends to be documented on paper, and to use manual procedures to control the flow of design and information between people and organizations.

Design management (also called workflow management) tools are commercially available from EDA suppliers and other commercial companies. Each has particular functional strengths and capabilities in terms of system platforms supported, ease of use, ease of integration of tools, checking, reporting, etc. Some are better adapted to the suite of tools supplied by the EDA company marketing the design manager while others are more generic and market support for EDA, Mechanical and Manufacturing engineering design.

Many potential customers of such systems foresee the need to use multiple design managers within their overall design system (unique requirements of specific engineering design domains, local preference at different geographic locations, etc.) while others are exploring internal proprietary solutions as a result of their needs diversity. Research in this field is exploring the addition of artificial knowledge based decision management capability that may someday provide automation to the design trade-off decisions engineering must make in today's complex designs.

In the current environment or state of the art for design management systems it is clear that:

- The formal definition of a complete design methodology is a complex, labor intensive task and once completed, it is not likely to be redone for the sake of reformatting it
- Customers desire the ability to use multiple design managers, and to change design managers with minimum impact should a "better" one become available (i.e., customer choice of "best practice")
- Many potential customers of these products require the ability to move design information from one design manager to another as the design progresses

Large enterprises are at various stages of recognizing the need for formal design management systems. Some, such as in the RASSP program, are exploring the personalization and use of commercially available offerings, while others are doing internal research and development of proprietary solutions. Most design teams interviewed, however, admit the need for formal design management systems to meet future design problems and objectives.

4.6.2 Requirements

The importance of automated design management is a function of the complexity of the design process (number of different process steps, or number of design elements, for example), the number of engineers or engineering teams involved in the design, and the required interactions between the different design domains required for successful design.

Based on the following technology trends, the need for automation to aid in design management will steadily increase:

- Design complexity will increase

  The number of design process and checking steps that must be successfully enacted to

complete a design will increase.

- Design team size (i.e., the number of engineers) involved with a single design entity will increase
- Design interactions across multiple design domains will increase

  Interactions between design engineering and manufacturing engineering in order to design for manufacturability will increase in importance as more sophisticated packaging evolves as will interactions between other domains such as electrical and mechanical design for PCAs, electrical and software design, etc. These lead to the following requirements.

4.6.2.1 Design Methodology Process Description Standard

The creation of a complete design methodology process description is a large and difficult task. Diverse groups within a single design enterprise need to use the same design process, but may need to use (or change to) different design management systems. It is imperative that a standard for representing this important design data be developed and adopted to enable design methodology to be portable across compliant design managers. Further, the overall business process may demand the use of multiple design management systems for a single business process, and potentially even multiple design management tools working on the same process.

Therefore, it is necessary that a standard for design process definitions be developed to meet the requirements for a portable and sharable process description.

This would support the need for multiple design managers within an enterprise and the ability to change design manager products without the cost of recoding the formal design methodology. This interchange standard must not impose restrictions on or create standard methods for entry and editing of the customer methodology.

4.6.2.2 Standard Interface between Design Tools and Design Manager

Potential customers of design management systems require the ability to encapsulate and execute design tools of choice (from multiple companies or locally developed) into the chosen design manager, without the need to modify the tools.

Therefore, it is necessary to develop either a standard interface between the design management tools and the CAD tools to support tool activation and deactivation, and that interface must be independent of and impose no specific requirements on the CAD tools.

Because business processes may cross multiple hardware-operating system platforms, the standards developed must apply equally well across UNIX, Windows, and other popular platforms.

4.6.3 Recommendations

There are a number of industry organizations who are or have been involved with design management standards including CFI, OMG, and the WfMC. Government funded programs such as RASSP and NIIIP have placed a high degree of importance on design management systems to their success and so, are encouraging the promotion of standards to meet the stated requirements.

Use of ITC message passing technology and standard messages is recommended as the standard interface between design managers and tools. Therefore, it is recommended that a set of ITC messages be developed and adopted by industry to meet the functional requirements for design management including tool launch, tool completion reporting, metrics gathering, etc.

4.6.4 Roadmap - Design Management

Efforts to develop standards for workflow managed environments and a portable design methodology process description should be accelerated.

4.6.4.1 Design Methodology Process Description Standard

It is imperative that a standard for representing design methodology process information be developed and adopted to enable design methodologies to be portable and sharable across compliant design managers to meet the above requirements.

For development of the methodology definition standard, it is recommended that the Industry Council commission an industry organization to define the requirements for this capability and issue an RFT to industry in order to build this standard off of an existing industry proven base. The final result of this activity should then be offered to a formal standards body (such as IEEE) for life-cycle support.

This item must be addressed with medium priority in the short term.

4.6.4.2 Standard Interface between Design Tools and Design Manager

A standard ITC-based messaging interface between the design management tools and the CAD tools must be developed to meet the above requirements.

It is recommended that the Industry Council commission CFI to work with the above organizations to develop and promote a comprehensive set of ITC messages to meet the functional needs of design management, and that CFI (or the X-Open consortium) provide a mechanism for life-cycle support of these messages. It is further recommended that OMG be commissioned for a parallel set of class objects.

This item must be addressed with high priority in the short term.

4.7 Design Data Management

Management of design data is an essential practice for electronic systems design at all package levels and across all system level design and detailed design phases. Management of design data (files and databases) necessarily includes the following elements:

- Data access information (user name to datafile name mapping)
- Data ownership and access authority information
- Managing concurrent data access
- Managing different levels (versions, iterations,..) of design data
- Design configuration (collecting all required data at the correct level for a design)

With increasing design sizes, projected increases in the number of designers involved with a design entity, concurrent design and globally dispersed design teams, the need for more formal and automated methods for data management (and using multiple data management subsystems) is expected to increase in importance and provide a growing opportunity for commercial data management products.

4.7.1 Current Environment

While commercial data management products are available from both EDA vendors and other commercial companies, they are often incorporated with proprietary solutions or other commercial products. This seems to be necessary in order to cover the total scope of data management requirements across large enterprises in particular. While all of the data management related commercial products offer strengths in one aspect or another, it is most often the case that no single product yet meets the total set of needs foreseen by the design community. On the other hand, all of the commercial data management products offer a basic set of services which are essential to data management. Thus, there is a great deal of functional overlap between competing products and certain value added features within each.

The current environment causes a number of problems to the potential customers and may actually be limiting the growth of commercial purchase vs. proprietary development. The specific problems in the current environment are:

- No standards for common data management functions

  There are no agreed to standardized terms for the basic service functions of data management systems. The definition of commonly used terms such as check-in, check-out, open, purge, update, submit, merge, retrieve, release, promote, version, level, workspace, etc. is open to interpretation, as is the function performed when any of these data management terms are called out for execution. This makes it very difficult for potential customers of these products to evaluate the merits of the value added features because they often cannot get an accurate understanding of the base on which they are built. Furthermore, the complexity of integrating commercial solutions together or with internal solutions is compounded by these “semantic” differences and often to a point where an internally developed solution is more economical.

- No standards for metadata representation

  Often, in large enterprises, there is a need to move data from one data management system to another as a design progresses between engineering domains or departments or geographical locations. This requires that the metadata (data records used by the design management system to manage the design data) needs to be extracted from one design data management system and reformatted into another. This mapping needs to be done in a fashion that guarantees no loss of information about the design; however with conflicting and ambiguous terminology used between data management systems, this task is far more complicated than simple data reformatting. Further, the completeness and accuracy of such translation is suspect, again causing potential customers to favor internal solutions.

Design data management is an extremely complex problem when one considers hundreds or thousands of design files, at different levels and versions, and which must be correctly interre-

lated to assure accurate manufacture, concurrent use of design data by multiple people and organizations, and distribution across tens or hundreds of computers. Further, these complexities are growing in direct proportion with the electronic technology advancements. That being the case, design groups would desire the ability to purchase design data management solutions from professionals well versed in the technology as opposed to internal investment into ad hoc solutions. However, the ability to do this is severely hampered by the lack of basic consistency standards as described above.

Making matters even worse, there are a number of disjoint consortium or government funded efforts that are seeking to rectify some of these problems by “standardizing” selected elements of design data management, including RASSP, NIIIP, STEP, WfMC and, undoubtedly, others. These disjoint activities will compound the problem even further for the EDA industry as they will, no doubt, each create their own set of standards which will be different from the other well intentioned efforts.

4.7.2 Requirements

Below are the key requirements for the design data management category.

4.7.2.1 Standard Interface between Design Tools and the Data Managment Subsystem

Any solution for design data management must be available as a design service that is independent of the approach to implementing design methodology, the EDA applications, and design files being used. Customers must be able to apply the chosen design data management solution on their desired design system of tools and must be able to do so without imposing restrictions on those tools and without the requirement for changes to those tools.

The CAD integrator must have the ability to call out design data management services from the user environment being used. This includes design methodology workflow managed environments as well as more manual design flow management using extension languages such as script file languages. In addition, it must be possible for EDA tools to choose to integrate such services without being required to do so. This ability must be available in such a way so as to allow for the customer to have the ability to choose or change the design management system as he/she would for any design tool.

In order to support the above, it is necessary to agree to a standard definition of the meaning and action of common design data management services. For each of these, an ITC Message should be defined and adopted to support the invocation of these services independent of the chosen management system and to support replacement of the design data management system without necessary change to the design environment or tools.

4.7.2.2 Design Data Management Metafile Interchange Standard

In order to support both the ability to incorporate multiple design data management systems within an enterprise, and the ability to substitute different design data manager(s), a standard must be developed and adopted for metafile data. This is to impose no requirements on the design data files, but only on the metadata records used within design data management systems.

4.7.3 Recommendations

To address the requirements for a standard interface between design tools and the data mangement subsystem, it is recommended that an ITC Message be defined. This would support the ability to call for the invocation of these services independent of the chosen management system and to support replacement of the design data management system without necessary change to the design environment or tools. EDA tools could optionally exploit these services.

It is also recommended that the Industry Council commission an organization to work with industry to develop the metadata standard.

4.7.4 Roadmap - Design Data Management

Below are the key roadmap items for design data management.

4.7.4.1 Standard Interface between Design Tools and the Data Management Subsystem

It is recommend that the Industry issue a “call to action” to companies and industry groups involved with design data management to establish the semantics and definition of the fundamental design data management services. It should be possible for this to be quickly developed using as the base those definitions currently in use by commercial offerings and within the industry standard groups (RASSP, NIIIP, STEP, WfMC). When complete, this standard set of function definitions should be offered to a formal standards organization (such as IEEE) for life-cycle support and dissemination across the industry.

CFI, or a similar industry consortium, should be commissioned to develop through its membership, the standard ITC Message set for these base services and provide a mechanism for life-cycle support. This could be accomplished either through its own organization, the formal standards body accepting the definitions, or the X-Open consortium. NIIIP and its affiliation with OMG should be commissioned to develop the Object oriented class members equivalent to the ITC messages with OMG as the preferred life-cycle support organization.

This item is high priority for the short term.

4.7.4.2 Design Data Management Metafile Interchange Standard

In order to support both the ability to incorporate multiple design data management systems within an enterprise, and the ability to substitute different design data management subsystem(s), a standard must be developed and adopted for metafile data to meet the above requirements.

It is recommended that the Industry Council commission an organization to work with industry to develop the metadata standard.

This item is low priority in the short term.

4.8 Design Process Metrics Management

Design process metrics management encompasses the collection, analysis and use of metrics about a previous design process in order to predict the needs of, and improve, a current design.

4.8.1 Current Environment

Current methods for design process metrics management are necessarily ad hoc with little or no support from the EDA industry. Design groups are finding it increasingly necessary to gather and analyze process metrics about CPU usage requirements, design data storage requirements, memory requirements and the like in an attempt to predict their capital requirements for the semiconductor technology explosion. Lack of any standard methods, and perhaps other business decisions, result in little or no customer support in this area from the EDA suppliers.

Inconsistent use of quantification terms such as simulation “events” per second, bytes of storage per “block”, etc. make it very difficult to characterize design tool requirements relative to design densities. Inability of tools to report in a consistent manner information about CPU and memory utilization for example, make it a customer burden to develop ad hoc facilities. Lack of standards for metric collection make it impossible to use this data in an automated manner (such as within a design manager) without custom internal development. Lack of rigid metrics for overall design productivity (such as circuits per person month) make it near impossible to analyze opportunities for improvement based on the history of other design projects. Though this is not perceived to be a show-stopper, it is a common complaint voiced by design teams who feel that design productivity will need to improve greatly in order to realize predicted future design densities in products.

4.8.2 Requirements

4.8.2.1 Standard for Design Process Metrics

It is highly desired that the EDA industry define a standard definition for a set of meaningful design process metrics that spans all design disciplines and design phases.

4.8.2.2 Standard Interface to Metrics Collection

It is desired that a standard interface to metrics collection be defined and supported by all commercial EDA tools so that standard process related information can be gathered in a standard way for analysis by design teams and or used within design management systems.

4.8.3 Recommendations

In the area of standards for design process metrics, it is recommended that the current work in the industry be combined under an Industry Council endorsed group, and that standard terms and definitions be created for use in metrics collection and reporting systems.

The recommendation to support standard metrics collection is to develop a standard set of ITC messages. This will enable the collection of various metrics based on capturing such design activity as tools being used, by which users, for what amount of time, etc.

4.8.4 Roadmap - Design Process Metrics Management

4.8.4.1 Standards for Design Process Metrics

It is recommended that the Industry Council endorse and support the activity currently underway at UNC (partly funded by SEMATECH) to develop EDA metrics. This work should be expedited with planned deliverables that can be adopted within the industry. In addition, it is recommended that the work at MIT under the RASSP program in this area also be evaluated and the results used appropriately.

This item is medium priority for the short term.

4.8.4.2 Standard Interface to Metrics Collection

It is recommended that the Industry Council commission CFI to develop a standard set of ITC messages and an approach for commercial EDA tools to report these metrics in a standard format.

This item is low priority for the short term.

4.9 Design Tool Management

Electronic design tools and systems have evolved rapidly over the last decade (though not as fast as product technology itself--hence the “technology gap”). Change will be a *constant* part of life for designers, the EDA industry, and the CAD system integrators. New requirements such as those in the SIA and IPC Roadmaps will drive the development of new tools that are not even envisioned today. New design tool technology must be inserted into the design process as soon as it is available in order to meet leading edge technology demands. This changing environment places a challenge on developers to provide tools that are easily installed and encapsulated into the environment(s) chosen by the designers and CAD integrators.

4.9.1 Current Environment

Using EDA design tools today is problematic in that each tool has its own, often unique, requirements for proper execution. Having to remember the installed tool location, executable name, command syntax, and required and optional arguments necessary to invoke a tool often leads to errors and decreased productivity. In addition, a given tool may require execution from several different environments (e.g., launch tool from the graphical desktop, or from a workflow manager). Today, encapsulation and integration of tools is unique for each tool in every environment. Development and maintenance of such personalized tool integrations is costly and error-prone, and can lead to longer time-to-market for the electronic product developers.

Therefore, there are several aspects of tool management that must be considered. The first is the management of individual EDA tools, that historically have been limited to providing a set of services to the individual design engineer. These services include the launching and stopping of tools, encapsulating the required inputs and outputs for a given tool, and grouping tools into logical clusters of tools (e.g., those tools used in design verification). Many of the EDA frameworks available today provide these capabilities. Also, managing an EDA tool has often been viewed from the perspective of a single user. However, in today’s environments, tool management must be viewed from a more global perspective. Larger design teams are working with a greater number of tools, spread over a larger geographical area. As the com-

plexity of designs grows, larger design data sets will exist across a larger number of machines. Tool managers must also be able to manage and track the status of tools on remote hosts and in heterogeneous environments.

The second aspect of tool management is associated with the management of computer systems. The rise in complexity of the design problem has lead to an increase in the number of design tools used by most users. In large design environments, installing and managing these tools has become a problem large enough to impact designer productivity. The methods for installing, licensing and upgrading tools differs with each vendor, and is often inconsistent even between tools from a single vendor.

4.9.2 Requirements

This section summarizes the requirements for inter-tool communication, tool encapsulation, and system management guidelines.

4.9.2.1 Standards for Inter-Tool Communication (ITC)

To adequately manage tools, a Tool Manager must have the ability to start, stop and determine the current status of individual tools. One way to accomplish this would be to build a monolithic environment containing all the tools the manager wishes to control. However, this scenario does not provide the flexibility needed for today's multi-vendor environments. A more efficient alternative is to provide a means of communicating between tools, via dynamic inter-tool communications. Since users need to manage tools from a variety of vendors, a standard message passing mechanism together with a standard message dictionary are required.

4.9.2.2 Standards for Encapsulating Tools

A tool encapsulation standard is required to capture the structural aspects of a tool. These include the tool's executable name and location information, tool class, operating environment requirements, and input and output requirements (along with their data types). It must be possible for this standard to enable highly automated encapsulation of tools into multiple usage environments including

- graphical user interface environment (e.g., CDE)
- workflow management environment (e.g, a workflow manager engine)
- vendor-specific environment (e.g., a framework)
- non-graphical environment (e.g., tool execution via script files).

For example, in a graphical environment, the encapsulation of the tool may result in the user being presented with a dialog box and asked to enter information such as the location of a design object. To enable encapsulating the structural aspects of a tools (e.g., inputs, outputs, launch mechanism), a common tool encapsulation method is required. This standard must support tools that operate in either graphical or non-graphical environments.

4.9.2.3 Standards For License Management

Because of the number of tools required from different companies and the sheer volume of licenses to be managed across multiple designers and computers, there is a requirement for

standard(s) for license management. Ideally, it is desired that all EDA tools use a single license manager and that all use it in a consistent way so as not to require the customer to create and maintain multiple key files and processes.

4.9.2.4 EDA Systems Management Guidelines

Another aspect of tool management is from the system management perspective. The rise in complexity of the design problem has lead to a proliferation of EDA tools that are often aimed at specific problems. In many user environments, the number of tools has increased dramatically, and will continue to increase. As a result, managing the installation and licensing of these tools has become a problem large enough to impact designer productivity.

Users need advice on the best methods for managing their EDA systems. While a standard may not be appropriate, a set of guidelines and best practices should be collected and shared with users. This document could also identify additional opportunities for standardization. For example, the method for installing tools differs with each vendor, and is often inconsistent even between tools from a single vendor. This document could recommend a common tool installation procedure be developed, such as the one used by tools in the Windows environment.

4.9.3 Recommendations

A number of vendors have implemented messaging facilities, that for the most part are used within a single vendor's tool set. The ToolTalk messaging facility, included as part of CDE, is vendor independent and has been endorsed by CFI as a standard inter-tool messaging mechanism. An additional advantage of ToolTalk is that the software will be provided by hardware vendors as part of their OS offerings. This removes the burden on tool vendors to implement another messaging facility. This standard must be made available on both UNIX and the Windows-PC environments.

However, using a common messaging facility by itself is not enough. Equally important, is a common set of messages, tools can use to communicate. CFI has developed a draft EDA Message Dictionary standard that should be used as a starting point.

With regard to standards related to tools integration, the CFI Tool Encapsulation Standard (TES) is the only existing standard in this area. TES has been evolving to meet EDA industry needs, and has recently been extended to better support automatic encapsulation of tools into the CDE environment, and to meet requirements such as those identified above. It should become standard operating procedure for EDA vendors to ship TES files with their products. All other tool creators and CAD integrators should also use TES to aid in integration of tools into the design process.

In the area of license management, it is recommended that the Industry Council commission the creation of an RFT for a commercially available license manager that meets the EDA community functional and business needs. Since the industry has more or less already converged on a single license manager product, it is anticipated that the functional capabilities of that product be adopted as a base standard, and modified only where there is a specific and justified EDA industry need (e.g. cost). In addition, a consistent policy for it's use must be developed. Further, it is recommended that upon selection of the winning RFT bidder that the

Industry Council do whatever it can to enforce the industry-wide use of this product.

It is recommended that the Industry Council commission a team of EDA companies to develop a standard use model for the winning license manager product so as to provide a consistent license management process for EDA customers independent of their tool choices. Life cycle support for this standard use model could be commissioned to EDAC or another industry organization of choice.

For the system management aspects of tool management, we recommend a set of guidelines or best practices be collected. The guidelines should cover items such as the tool installation, file location strategies, license setup, and managing multiple released versions of a tool. Guidelines should be available for both the UNIX and Window-PC environments.

4.9.4 Roadmap - Design Tool Management

This section outlines the recommendations for inter-tool communications, tool encapsulation, and systems management guidelines.

4.9.4.1 Intertool Communications: Adopt ToolTalk, Provide Windows Interoperability

The recommendation is to adopt ToolTalk as the standard for communications between tool managers and EDA tools. ToolTalk together with a standard EDA Message Dictionary can meet the requirements for inter-tool communications. The ToolTalk messaging facility should be supplied by UNIX hardware vendors as part of their OS offering. It must also be made available on the Windows-PC platform. Since the Message Dictionary is EDA specific and its contents will change over time, it should be developed and maintained by an EDA organization such as CFI. Also refer to 4.2 "Design Tool Communication" for additional discussion on this topic.

This item should be adopted as high priority immediately (i.e., in the short term).

4.9.4.2 Tool Encapsulation: Adopt CFI TES

Since CFI TES meets all the documented requirements for support of tool encapsulation into the various EDA desktop, workflow management, and vendor-specific usage environments, it should be immediately adopted. Standard TES files should be shipped by EDA vendors at their next major release of their tools to enable the automatic installation of new tools into the above EDA environments, and to minimize impact on CAD integrators when new tools are installed for use.

This item should be adopted as high priority immediately (i.e., in the short term).

4.9.4.3 License Management: Establish Base EDA Industry License Manager and Use Policy

While there are some reasonably effective license management technologies available today to support the management of design tool software, the policies for their use are not standardized and cause much consternation in the user community. This task is to establish a common EDA industry-wide tool license manager, and a common policy for it’s use.

This item should be adopted as high priority in the short term.

4.9.4.4 EDA Systems Management Guidelines

The recommendation for this item is to collect a set of guidelines or best practices to advise users with regards to EDA systems and tool management. This document should be created by users working closely with the tool vendors, and owned by a user group such as USE/DA.

This item should be adopted as low priority immediately (i.e., in the short term).

4.10 Resource Management

With the advance in technologies, the design complexity is increasing at a compound rate of over 50% per year. The SIA roadmap predicts that the design team size will approach 270 in 1998 and 600 in 2001. These large design teams will require an environment of hundreds of networked desktops and servers, a large number of Electronic Design Automation (EDA) and productivity tools and, millions of data files residing on tera-bytes of disks.

In the future, the EDA users will face two key problems:

1. How to access and manage these resources (machines, disks, licenses, data...).
2. How to efficiently utilize these resources.

“The challenge is managing the exploding complexity in the design environment”

4.10.1 Current Environment

The current design environment consists of heterogeneous hardware platforms and software packages. The different hardware and software generally do not communicate with each other and usually an internal CAD group has the charter to make all of these components work together. Although the hardware is networked together, access to one machine is very different from another because of the operating system, directory structure, license server, and so on. To overcome these difficulties, the CAD group develops “glue” software to link the different software packages to form automated design flows to be used by the designers. Further automation is customized by individual users through the creation of “ad hoc” scripts.

The design environment resulting from these “band-aid” solutions is very fragmented and highly individualized. Automation can be achieved at the individual level, but there exists no mechanism to share data or tasks at the workgroup level.

Standards will be greatly beneficial in prescribing uniform ways, best practices or guidelines to organize and access the resources in a design environment.

4.10.2 Requirements

A study on the ratio of computers to users over time reveals, that in the future, each user will have at his or her disposition a large amount of computer resources: the challenge is how to harness these resources. As indicated in graph 5.3.5.2.1 in the seventies, a group of 50-100 people share a mainframe for their computing needs with a ratio of 1/50-100. In the eighties and with the introduction with the workstation, most of the users have a dedicated computer and the ratio becomes 1/1-5. In the nineties, the hardware cost has significantly dropped and the design complexity has increased dramatically resulting in a ratio of 2-5 computers for each

user. This trend is expected to continue in the year 2000 with 10-15 computers available to each user.

The requirement is to transform the hundreds of networked computers, and the corresponding data and tools, into a single and “virtual” resource pool for the users.

These resources include: computers, disks, networks, tools, and data. They need

to be tightly coupled with each other in order for the users to achieve the full benefit.

4.10.2.1 Support for Load Balancer/Job Scheduler

A Load Balancer/Job Scheduler will manage the queueing of the jobs and balance the load of the computers both for interactive and batch jobs. This facility will allow the user to efficiently utilize all available computer resources. The user should be able to dynamically reconfigure priorities based on job types, user profile, project, time, and machine configuration.

4.10.2.2 Support for Uniform Way to Organize the Design and Related Data

A uniform way to organize design data and its hierarchy, will present users throughout the enterprise with a uniform view independent of which computer they log on. This feature is important since users and the jobs they create, can access any node in the network and an consistent view of the data is a must.

4.10.2.3 Support for Workgroup Design Data Management

The design data is the most important resource in the design environment. There must be a facility to manage design data through the design cycle, and its evolution through the workgroup.

A release and archival facility for the design data is needed.

4.10.2.4 Support for “Design Warehouse”

In the complex circuit development environment, the user is presented with a large amount of information to be analyzed, in order to make decisions and to take action. A “design warehouse” for the design data, is recommended to facilitate this task. The “design warehouse” would collect, sort, organize, and present data in views appropriate for the different user classes. Based on predefined rules, the users will be presented with options on how to proceed.

4.10.3 Recommendations

There are no specific standards activities recommended at this time. However, the above areas should continue to be monitored regularly as technology evolves.

4.10.4 Roadmap - Design Resource Management

There is no specific standards roadmap for this area at this time.

5 The Design Information (Design Data Representation)

This chapter includes descriptions of important design information (data) generated and/or used in each of several key activities related to the electronic design of components and systems as described below.

The design information environments for designing cores and chips, MCMs, and boards are similar in many ways, as are the requirements for future support. Environmental comments, unique to a type of packaging, are identified and discussed within the relevant design category. Therefore, after some debate, it was decided that for the purposes of discussing the relevant EDA standards, it would be appropriate to discuss the design phases in two major categories called “system design” and “detailed design”. This is not meant to imply a particular design process or methodology, but rather to enable groupings of related standards. These environments and some rationale for this grouping are described in some detail below.

In order to eliminate redundancy, requirements that were common across each of the above categories were grouped together in the section “common topics”. At the end of this chapter, the technology rules and models required to support all phases of design are discussed in the section on “design technology re-use”.

Therefore, this chapter is organized as follows:

- 5.1 "Common Topics Across All Design Information"
- 5.2 "Common Topics Across All Design Steps"
- 5.3 "System Level Design"
- 5.4 "Detailed Design"
- 5.5 "Design and Technology Re-use".

5.1 Common Topics Across All Design Information

The topics discussed in this introductory section are important independent of the design step involved, and also independent of the design information represented. The topics represent additional requirements to the unique requirements stated in the following sections (5.3.4 "Roadmap - System Level Design" and 5.4.4 "Roadmap - Detailed Design"). They are stated here in order to be more concise, and avoid repeating this information in both sections.

5.1.1 Incremental Processing

This section addresses incremental processing: the current environment, requirements, recommendations, and roadmap.

5.1.1.1 Current Environment

The rapid growth of design size as indicated in the SIA Roadmap implies a corresponding growth in the amount of design data. During the execution of the design process, design engi-

neers make small changes in the design (small relative to the entire design), and then the effect of those changes is evaluated.

What appears to be a “small design change” by the engineer actually affects a “large file” (in the case of a large SDF file or EDIF file, for example). The amount of time needed to reprocess these large files, is becoming prohibitive, and is not in proportion to the designer’s changes.

5.1.1.2 Requirements

5.1.1.2.1 Support Incremental Processing in the Design Representation Standard

This situation drives a need for standards that support the relatively high performance requirements for interactive programming interface communication between EDA design tools. For example, the interaction between logical and physical design tools when performing timing verification on a design. This requirement is related to 4.2.4 "Roadmap - Tool Communication", but has EDA domain-specific connotations (e.g., message passing of items such as “highlight net”).

There is *also* a need for incremental file-based communication for logical connectivity and physical design data. This type of incremental file-based approach is needed to support the transfer of design data between environments where file sharing or inter-tool communication is not possible, feasible, or practical. An example of where this is a requirement, is the interface between an OEM customer and an ASIC design shop, where direct data sharing is not possible. The information needs to be transferred, and exchanging messages between tools may not be practical. In such cases, file-based transfer may be a preferred method of moving the design data to where it has to go. For additional justification of the need for incremental processing of design data, refer to 5.4 "Detailed Design", and specifically Table 5.1: "Impact of Design Size on Design Processing Times".

There is a clear requirement for design representation standards such as 5.4.2.1 "Standard Detailed Design Representation" to support incremental processing of design changes (or engineering changes). Incremental processing as used here applies to the entire design cycle and any phase within the design cycle. This requirement needs to be implemented via appropriate programming interface and file-based solutions to meet the total set of requirements.

Support for incremental processing must be architected into each of the following areas:

- 5.4.4.1 "Converged Industry Standard for Logical Connectivity"
- 5.4.4.6 "Standards to Support Floorplanning"
- 5.2.1.4.1 "Integrate Timing Information into Design Representation Standard"
- 4.6.4 "Roadmap - Design Management".

5.1.1.3 Recommendations

It is recommended that support for incremental processing be designed into the proposed design representation standards. Refer to 5.4.2.1 "Standard Detailed Design Representation".

5.1.1.4 Roadmap - Incremental Processing

5.1.1.4.1 Support Incremental Processing in the Design Representation Standard

Incremental processing must be supported in 5.4.4.1 "Converged Industry Standard for Logical Connectivity" and the follow-on standards. The base standard and any extensions should be monitored so that the strategic standard can include the necessary information. EDA vendors should support incremental processing of the strategic design representation standard as soon as it meets basic requirements.

This item should be adopted as high priority immediately (i.e., in the short term).

5.1.2 Hierarchical Processing

This section addresses hierarchical processing: the current environment, requirements, recommendations, and roadmap.

5.1.2.1 Current Environment

As was stated in 5.1.1.1 "Current Environment", the rapid growth of design size as indicated in the SIA Roadmap implies a significant growth in the number of designers on a design team. In order to manage very large designs, it is a common practice to break up the design into pieces so that the pieces can be designed independently, and yet be put together in a system design for system level processing. There is therefore a corresponding growth in the number of design objects being designed by the design team. This hierarchical design approach of “divide and conquer” must be supported.

5.1.2.2 Requirements

5.1.2.2.1 Support Incremental Processing in the Design Representation Standard

There is a requirement for design representation standards to support hierarchical processing of design changes (or engineering changes). Hierarchical processing as used here applies to the entire design cycle and any phase within the design cycle. This requirement needs to be implemented via appropriate programming interface and file-based solutions to meet the total set of requirements.

There is a requirement for EDA tools, independent of the design phase, to support hierarchical design representation and processing of design elements. This requirement has some similarities to requirement 5.1.1 "Incremental Processing", but is focused on handling design information across the design hierarchy.

Support for hierarchical processing must be architected into each of the following areas:

- 5.4.4.1 "Converged Industry Standard for Logical Connectivity"
- 5.4.4.6 "Standards to Support Floorplanning"
- 5.2.1.4.1 "Integrate Timing Information into Design Representation Standard"
- 4.6.4 "Roadmap - Design Management"

5.1.2.3 Recommendations

It is recommended that support for hierarchical processing be included in the proposed

design representation standards. Refer to 5.4.2.1 "Standard Detailed Design Representation".

5.1.2.4 Roadmap - Hierarchical Processing

Hierarchical processing must be supported in 5.4.4.1 "Converged Industry Standard for Logical Connectivity" and the follow-on standards. The base standard and any extensions should be monitored so that the strategic standard can include the necessary information, and EDA vendors should support incremental processing of the strategic design representation standard as soon as it meets basic requirements.

This item should be adopted as high priority immediately (i.e., in the short term).

5.1.3 Design Object Naming

This section addresses design object naming: the current environment, requirements, recommendations, and roadmap.

5.1.3.1 Current Environment

There are many useful design tools available that use naming conventions that have been developed to support legacy naming conventions for EDA design objects (such as net name, block name, etc.). In a multi-vendor design system environment, this frequently causes significant confusion, particularly when several tools that use different naming conventions are in a tight design loop.

Work in the area of name mapping is required to allow tools from different EDA vendors to map between the names they use for the same EDA objects. This capability is a prerequisite for doing many operations (such as cross-probing).

Most of the design objects that EDA tools want to access have a “name” attribute. However, in a given vendor’s design tool or system, this name may have limitations (e.g., on the length or character set that can be used for a name). Further, the specific rules for naming objects can vary from application to application, and from vendor to vendor. This creates a very complex mapping problem in today’s multi-vendor environments and confirms again the difficulty of writing translators to convert files from one form to another. This problem is exacerbated when the information passed from one tool to another is in a key design loop where inter-tool communication is desirable.

5.1.3.2 Requirements

A standard for design object naming is desired in the implementation of the roadmap for the design data representation (see 5.4.2.1 "Standard Detailed Design Representation"). This standard should define the strategic design object naming convention, and support for legacy applications, so there are defined mappings to/from that standard naming convention to vendor-specific naming conventions. This naming standard must also meet the needs for communicating mappings of named objects via standard inter-tool communication mechanisms (see 4.2.2.1 "Standards for Inter-Tool Communications").

5.1.3.3 Recommendations

It is recommended that a new standard naming convention be developed for EDA design

objects. It is also recommended that in order to support coexistence and migration to that new standard, a set of mappings from today's naming conventions be documented so that the specification can define how existing vendor applications naming conventions map to that standard.

5.1.3.4 Roadmap - Standard Design Object Naming Specification

Develop a standard design object naming convention for the next decade, and define how existing vendor applications naming conventions, map to that standard. Ensure that 4.2.2 "Requirements" and 5.4.2.1 "Standard Detailed Design Representation" will both be included in this work. Refer to those sections for
further information.

This item should be adopted as high priority immediately (i.e., in the near term).

5.2 Common Topics Across All Design Steps

The topics discussed in this introductory section are important independent of the design step involved, and are global requirements, that must be met, in addition to the unique requirements stated in the sections that follow (5.3.4 "Roadmap - System Level Design" and 5.4.4 "Roadmap - Detailed Design"). They are stated here in order to be more concise, and not repeat this information in both sections.

5.2.1 Timing Information

This section addresses timing information: the current environment, requirements, recommendations, and roadmap.

5.2.1.1 Current Environment

Many current timing driven design tools interface to the defacto Standard Delay File (SDF) batch timing information file. The tools that access the SDF read the entire file and correlate the timing data, with separately specified connectivity or structural data describing the design. The existing standards (e.g., EDIF, CFI DR, AP210) do not yet (formally) include the timing data.

This approach will not be adequate for the large designs of the next decade, or even for the more immediate term. These very large chips will require many designers on complex chip designs and system designs and those designers may well be spread across geographic regions. The delay due to wiring will be 80-90% of the delay in 1996 in .35um technology. Complex large high-performance chip and system design, often requires the inclusion of timing design as an embedded part of the early high level design. Timing has become a key constraint in constraint-driven design.

Integrated Environment

In an integrated environment, where structural design changes or wiring changes can be made, the engineer will need the delay calculation and timing analysis to be done on only the changed area (or design object). The engineer will want to see the results back annotated on

the source form. This will require that the timing data be available in the procedural interface form on a hierarchical and incremental basis, and across the geographically dispersed network.

Separated Environment

In a separated environment, such as an OEM designer interfacing with an ASIC foundry doing the back end design, the OEM designer will need the capability to interface floorplanning timing constraints, and timing estimates, with the structural data. This data is the basis for the designer's initial verification. As hierarchical and incremental processing is added across this interface, the OEM designer will need the capability to transfer incremental timing changes that correspond to the incremental structural changes in an integrated fashion. After physical design is completed, the final timing data must be sent back to the original OEM designer, without requiring that the 10's of megabytes of structural data be included.

5.2.1.2 Requirements

5.2.1.2.1 Timing Information in Detailed Design Representation

There is a requirement for design representation standards such as 5.4.2.1 "Standard Detailed Design Representation" to support EDA tools so that they can share and/or exchange timing information, in a standard inter-operable way across all design phases.

Each EDA timing driven design tool, should be driven from, and constrained by, the timing performance and implementation requirements of a design. There needs to be a consistent architecture for the following:

- timing design information entry
- timing constraint capture
- timing analyzer use
- high level early interconnect estimation
- manipulation of cell hierarchy and timing libraries
- timing information storage
- timing design management and change control
- other design activities using timing driven design information.

During the early phases of the design, timing may be a matter of re-use information, area based estimates, or other design requirements. Later, after the physical design, timing is fully determined by the structural, behavioral, and physical detail of the design. The timing driven design process requirement, therefore, is embedding the timing design into the usual mixture of Top Down, Bottom Up, and Middle Out design process paradigms.

- Top Down - Timing needs to be imbedded in the design of the overall target system functions and performance targets.
- Middle Out - Timing needs to be propagated upward to higher levels of design and downward to more detailed level of design.

- Bottom Up - Timing needs to be imbedded in the early design of the underlying chip technology, low level design functions, and cell and transistor libraries used to implement the low level design functions. This requirement includes interfaces to technology design for joint development of accurate models. These lower level timing parameters need to be abstracted, and propagated up the hierarchy.

Once the initial design steps have been completed, timing design must continue to be embedded in the incremental design of the hierarchical chip.

These requirements include support for timing representation so that static timing analysis and other calculations, can be performed hierarchically on a design element, without reprocessing an entire design to evaluate a change in only one element in the hierarchy. Some technologies have unique timing information representation requirements, such as the rise time dependent delay requirements of CMOS.

The following paragraphs identify information and supporting infrastructure that is required in properly support timing design:

- Timing Characteristic Type

  A timing characteristic type identifies the role of the timing information in the design. The specific terms included in this class should, include timing characteristics (such as given in the DIE-Timing latest draft document) including delay, setup, hold, pulse width, cycle time, period, rise and fall time).

- Application Timing Information

  Application timing information identifies the designer's intended purpose, or origin of a set of timing values. An important set of application information to include are those that identify timing design management information, (e.g., required, budgeted, analyzed, actual, measured, and current). Each of these can be presented in a statistical manner with nominal, mean, min-max, sigma, confidence level, skewness, and kurtosis.

- Variational Timing Information

  Variational timing information provides a means for defining timing information values as a function of other design requirements, constraints, and parameters. The expressive capabilities should encompass those of the Delay Calculation Language (DCL). The expressive capabilities should include functionality required to support physical based delay estimate calculations, including high level abstract floorplanning. Capabilities should also include a facility for defining the sensitivity of timing information to design process parameters and value set changes. For example, an ability to determine the sensitivity (rate of change) of delay to design and process parameters.

  The above items will provide the capability to calculate delay using varying electrical parameters. These electrical parameters include source and load information that may be associated with circuit transistor, switch, gate and block levels of design.

- Interconnect Net and Path Identification

  Means should be provided to identify the timing design related function, status, and criticality of the elements of net and path interconnect within a hierarchical chip design. Iden-

tification should include facilities such as nets, net segments, paths, ports, and signals. The application function of the interconnect should be identified, (e.g., clock, bus, power, and ground). The criticality of the interconnect should be identified as critical or noncritical.

5.2.1.2.2 Delay Calculator Language (DCL)

The requirements in support of the proposed standard Delay Calculator Language (DCL) are as shown in the Standard Delay System Objectives Version 1.1, 11/8/94:

- Allow silicon foundries to describe delay equations for ASIC libraries, and in a single way which can be used by any set of CAD applications and vendors
- Allow CAD vendors to interface to delay calculation for specified design elements using a single interface to the foundry supplied equations
- Account for wire capacitance as well as gate capacitance, which implies more complex equations; more CAD tools are now required to access delay calculation with these complex equations
- Provide a Delay Language compiler which creates a compiled form of the delay expression language, which when executed with a specific net description, can calculate all necessary delay characteristics of the net and associated gates. This then provides the *same* set of delay values to each vendor tool
- Provide a programming interface (PI) to a compiled form of delay equations described in the delay equation expression language
- Delay equations distributed from the semiconductor vendor must be protected from accidental or intentional loss of intellectual property rights
- The calculation from the delay equations must be of high enough performance to support both the batch calculation of all delays, and the incremental calculation of individual nets within design applications such as synthesis. This calculation is to include both pre-layout and post-layout phases of the design process

In addition to the above, DCL is required to support arrays and FOR loops to support transmission line analysis.

5.2.1.2.3 IBIS Standard Enhancements

The IBIS standard must be enhanced to support non-monotonic drivers. Lack of this support, is preventing whole technologies from supporting the IBIS standard, required for delay calculation and signal integrity above the chip.

5.2.1.2.4 Common Delay Calculation

A common delay calculation capability is required to provide consistent timing information across applications.

5.2.1.3 Recommendations

In the area of timing information in detailed design representation, it is necessary that the information transfer and correlation processing time of timing data be minimized. This is due to:

- the extreme significance of the wiring on the delay characteristics of the design
- the pervasiveness of the timing impact throughout the design process (i.e., in both system level design and in detailed design)
- the large systems being designed across geographic dispersed regions.

Therefore, the current defacto standards like SDF need to be integrated into the proposed standard design representation (both procedural and file-based interfaces). The transfer of timing data between functions should be supported, integrated with the structural data or independent of the structural data, to support the many different methodologies and design scenarios.

To meet the requirements in this area, it is recommended that

- Timing information be added to the detailed design representation standards described in 5.4.4 "Roadmap - Detailed Design"
- IBIS be enhanced to support non-monotonic drivers, to meet the above requirements
- The CFI/OVI Delay Calculation Language and PI effort be completed, and then extend this capability beyond ASIC packages.

5.2.1.4 Roadmap - Timing Information

5.2.1.4.1 Integrate Timing Information into Design Representation Standard

Timing Information currently supported by standards like SDF should strategically be supported by 5.4.2.1 "Standard Detailed Design Representation". Extensions to SDF should be monitored so that the strategic standard can include the necessary information, and EDA vendors should migrate to the strategic Design Representation Standard as soon as it meets requirements for timing information.

This item should be adopted as high priority immediately (i.e., in the short term).

5.2.1.4.2 Enhance IBIS Standard to Support Non-monotonic Drivers

The IBIS standard must be enhanced to support non-monotonic drivers.

This item should be adopted as high priority immediately (i.e., in the short term).

5.2.1.4.3 Complete the CFI/OVI Delay Project and Extend Beyond ASIC Packages

The delay project to demonstrate the applicability of DCL and the PI to delay calculation must be completed.

This item is high priority in the short term for ASICs, and medium priority for PCB and MCM packages.

5.2.2 Simulation and Test Control

This section addresses standards for simulation and test control: the current environment, requirements, recommendations, and roadmap.

5.2.2.1 Current Environment

There are a number of different simulators that can be used during the system level and detailed design phases. For most of these simulators, there is a unique way to specify the stimulus and expected response, control the simulation, control the debug of the design, and collect the results from the simulation. Moving this type of information from one simulator to another is a time-consuming, difficult conversion effort. The lack of a common simulator control language is an inhibitor to simulator “plug and play”.

In addition to the lack of a simulator control standard, there are also several test vector formats being used to drive testers in the manufacturing test environment. Moving test information such as stimulus and expected response, and other control information from one test environment to another is also difficult.

5.2.2.2 Requirements

Standards are needed in this area, that addresses both simulator and manufacturing test requirements.

5.2.2.2.1 Standard Simulation Control Specification

A standard for specifying stimulus/response and other simulation control information is required. This standard must encompass support for the stimulus and expected response, control the simulation, control the debug of the design, and collect the results from the simulation. It must address analog, digital, mixed analog/digital and mixed language designs.

5.2.2.2.2 Standard for Test Control Specification

A standard for specifying stimulus/response and other manufacturing test control information is also required. This standard must encompass support for the stimulus and expected response, control the test, control the execution of the test, and collect the results from the test. It must address analog, digital, mixed analog/digital and mixed language designs.

5.2.2.3 Recommendations

A new industry standard should be developed to support a common simulator control specification. This should be a single standard that addresses both the simulator environment during design, and manufacturing functional test requirements.

5.2.2.4 Roadmap - Standard Simulation/Test Control Specification

5.2.2.4.1 Converged Industry Standard for Simulation Control Specification

A new industry standard should be developed to address the requirements for a common simulator control language to meet the requirements stated above.

The IEEE DASC Simulation Control Language group intends to start paperwork for PAR submission (completion date before 06/96) on this Standard Simulation Language. The work of this group should be considered in the development of the industry standard.

This item should be adopted as medium priority immediately (i.e., in the short term).

5.2.2.4.2 Converged Industry Standard for Test Control Specification

The simulation control standard developed in the above step, should address not only simulator control requirements, but enable the subset of simulator control information (e.g., stimulus

and expected responses, and loop controls, etc.) to be included as part of the key interface to manufacturing. This would allow testing of the manufactured design object in the same way that the design was simulated during the design phase.

This item should be adopted as medium priority immediately (i.e., in the near term).

5.3 System Level Design

This section addresses system level design: the current environment, requirements, recommendations, and roadmap.

5.3.1 Current Environment

The system design step covers the architectural and high level design phases of electronics system design, and creates the high level design plan to be implemented in the detailed design phase that follows in the next section.

The architectural design phase is typically characterized by very high level design activities, leading towards the establishment of overall partitioning into physical packages, and technology selection for each of the major design objects of the product architecture. Design tasks include evaluating alternative architectures, assessing performance or throughput of candidate architectures, performing various hardware/software trade-off analyses, considering re-use of previously designed objects as part of the architecture, starting of hardware/software co-design, and considering life-cycle costs of various early design decisions. Architecture design and the relationship to software co-design is discussed in 6.4 "Software Design Interface". Examples of current related work in this area include SpecSyn and the Ptolemy project at the University of California, as well as in the RASSP program.

During the high level design phase, the functional details of each of the major design objects from the architecture phase are modeled functionally in system level design specification languages. The languages typically used include: VHDL, Verilog, C, or possibly a proprietary specification language. The entire system is evaluated via simulation or other functional evaluation tools to establish confidence that the architectural design concept meets key design function and timing objectives. As functionality is refined, preparation for the detailed design phases gets underway. High level floorplanning and synthesis can be used for chips to generate an implementation level design for the selected IC technology, and to generate or capture constraints on implementation.

Also during the system level design phase, test strategies such as BIST, traditional stuck fault, chip in place, and delay testing are developed.

5.3.2 Requirements

Support for VHDL and Verilog standard hardware description languages is required, as well as any future HDL representations for architectural and system design. A common interoperable interface from any HDL, is required to support independent EDA design tool selection and use. An example of this is the effort by the OMF to develop an HDL-independent simulator interface. Any simulator that uses this interface can simulate models written in any

HDL compliant to the simulator interface.

A key enabling technology for very large designs and designs that re-implement previously developed designs has to do with support for architecture design specification and associated re-use. As the popularity of architecture and high level design using hardware description languages such as VHDL and Verilog continues to increase, the potential for re-use of this work increases. Standards to support this are still emerging, and design representation technology needs to be established to encompass such information as design specification guidelines for encoding the various simulation models (architecture level, RTL level, implementation level, etc.), as well as functional test vectors, and other design data that was developed in the original version of the design. What is a chip today will only be a part of a chip (i.e., a “core”) tomorrow. What is a large subassembly today, will be a chip tomorrow. While this requirement is listed in the architecture phase, it applies to all of the design phases, in the sense that re-use of previous design information is required.

In addition to standards that support HDL specification of designs to support re-use, there is a challenge to enable existing legacy designs to be available for re-use in cases where there is no HDL specification. Tools and techniques for generating models of implementation level designs are needed to support re-use of such designs.

5.3.2.1 HDL Standards to Support Synthesis

While there may be compelling reasons (size of customer set, etc.) for having more than one design language (i.e., VHDL, Verilog, C), there are also some compelling arguments for having a standard set of synthesizable primitives, or subsets of HDLs, along with standard ways of constraint setting and passing of this information to all applicable tools in the Design Environment. The idea of a standard set of HDL primitives is attractive in that very high level design tools could target the HDL primitives as an output, (i.e., a design point from which design synthesis is effective, and synthesis tools could target the HDL primitives as an input, from which design implementation is effective). Standardizing on a common set of HDL primitives could provide a foundation for future innovations in systems design.

5.3.2.2 Standard HDL Interfaces to Design Analysis Tools

There is a need for various design analysis tools, including simulation (and other tools) to become more independent of the HDL chosen for doing design specification. There are two important scenarios with similar, yet distinct requirements:

- The “original design scenario”

  In situations where *original* design is being performed, design teams have a requirement to chose their preferred HDL in which to design, simulate, and eventually produce the design object. This is the typical scenario of the design team in a component supplier company, who have various design teams who may use different HDLs for a variety of reasons. They may choose VHDL or Verilog based on familiarity and experience or other business reasons. The same scenario is true in large system design companies where multiple HDLs are used for a variety of reasons. In any case, if the design object is reusable and becomes a building block for use in higher level design, there are additional requirements; for additional information on function reuse, refer to 5.5.2.3 "Standards for Reusable Functions". When designing a single design object, the design team chooses an HDL,

and the simulators which support it, and performs design.

- The “reuse scenario”

  In any design situation where teams desire to make use of previously designed and reusable design objects such as those from the “design scenario” above, the reusing design teams have requirements to also choose their preferred HDL in which to design and release this higher level of design object, independent from the HDL that any of the building blocks may have used. That HDL may be the same or different than the HDL used to design some of the reusable building blocks.

There are no specific requirements for VHDL and Verilog interoperability within a single original design scenario as described above; however, when in the reuse scenario, it is clear that the reusing design team must be able to choose their HDL and simulator independent of the choices of the original design team.

As extensions are made to HDLs (individually or in a concerted collaborative fashion), for example mixed digital/analog capabilities, there is a requirement that those extensions meet the above standard HDL interface requirement.

5.3.2.3 Standard Controls for Constraint Driven Design

There is a requirement for a standard for specifying design constraints on the design entity being developed. This promotes interoperability between tools such as synthesis and the HDL languages. Metrics required include specification of target values for total area, total power, maximum path length, specific path point-to-point timing, EMI, etc. In addition, the standard must include the ability to specify cell and power level, and specific physical location for large entities on the package (e.g., large arrays or microprocessors). This standard should be supported in a standard way so that HDL-independent controls can be established for synthesis and the subsequent detailed design activities.

5.3.2.4 Standards to Support Floorplanning

There is a requirement to share information between the synthesis and floorplanning activities performed during system level design. This data includes the specification of such information as cluster specifications, physical boundary requirements, and other floorplanning constraints. This information must also be shared with the detailed design phase so that early high level floorplanning and synthesis can be effectively linked with detailed floorplanning, placement and wiring, etc.

5.3.3 Recommendations

For system level design in general, it is recommended that a new effort be started to develop a system design standard (for use later in the decade), using the standards development process recommended in 2.3.2 "Standards Development Process Recommendations". This work should be started by building a system level design information model using VHDL as a guide. Once this is completed, Verilog coverage should be evaluated, and any required extensions made to the information model. This process will clearly identify where the information between the two languages is the same and where it is different. This approach also will facilitate adding new common information to the information model and the two languages in the

same way (such as a common constraint language, or possibly analog extensions, or support for higher architectural levels of design). To the extent that the information model between VHDL and Verilog overlap (i.e., the information being modeled is the same information), it should be observed that the PI to access that information is identical.

In the synthesis area, to support the concept of raising synthesis to a higher level, it is recommended that a standard set of synthesizable primitives among HDLs be established.

To address HDL independent design analysis, a promising approach is to complete the work of the OMF, whose goal is to have a simulator interface for compiled HDL models (be they VHDL, Verilog, C), and for the various languages to have a compiler to that OMF interface. This would support the concept of language neutral standard interface for simulators.

It is recommended that the work of the OMF be accelerated, that “negative delay” as proposed for VITAL 3.0 be supported at initial release of the OMF, and that an HDL-independent simulator interface be developed to meet the requirements of the design scenarios described above. It is further recommended that the planned OMF interface be later extended to include analog support (after VHDL-A and VERILOG-A are balloted).

Currently, the OMF effort is only targeted for HDL independence for simulation. Synthesis and formal verification tools must continue to support multiple HDLs until and unless additional approaches beyond the planned OMF become visible which could make those design tools more HDL independent.

For meeting the requirements of constraint driven design, it is recommended that standard keywords be established, usable in both VHDL, VERILOG, and any other design language, to specify target values. This would most likely be an ongoing collection of keywords much like a message dictionary. CFI could be the keeper of this standard.

It is also recommended that a standard be developed to support the floorplanning requirements described above.

5.3.4 Roadmap - System Level Design

5.3.4.1 Standards to Support System Level Design

It is recommended that the Industry Council commission a task group to begin work on a system design standard. As used in this document, system level design includes both architectural and traditional high level design. Innovations in architectural design are emerging for which new standards will be required. Since VHDL is significantly more comprehensive as a design language for higher level system design, the recommended approach is to start with VHDL, and build an information model. Then, information model coverage of Verilog should be tested (with the assumption that much of the information model derived from VHDL also has a relationship to Verilog). It is the goal that once these information models are developed (and extended for architectural design as required), a system design standard can be released. Potentially, use models for VHDL and Verilog could also be released.

This item is high priority for the near term.

5.3.4.2 HDL Standards to Support Synthesis

A standard set of synthesizable primitives among HDLs must be established. An IEEE workgroup is currently working on the problem of synthesizable subsets. The Industry Council should endorse this effort or commission a task group to address this problem.

This item is high priority for the near term.

5.3.4.3 Standard HDL Interfaces to Design Analysis Tools

The OMF effort described above is high priority in the immediate time frame including the support for negative delay in constraint specification as well as in normal path delay specification.

The analog extension is high priority *after* VHDL and VERILOG support analog. This item should be implemented as high priority in the immediate time frame.

In this area, efforts must continue to focus on improved interoperability of models from different sources (suppliers) to support system design and analysis.

5.3.4.4 Standard Controls for Constraint Driven Design

Standard keywords, usable in VHDL, VERILOG, and any other design language to specify target values for the metrics discussed above, must be established. It is recommended that the Industry Council commission a task group to focus on this work.

This item is high priority in the immediate time frame.

5.3.4.5 Standards to Support Floorplanning

A standard must be developed to support floorplanning requirements.

This item is high priority in the immediate time frame.

5.4 Detailed Design

This section addresses detailed design: the current environment, requirements, recommendations, and roadmap.

5.4.1 Current Environment

In the detailed design phase, the system level design created in the previous step, is transformed into detailed logical and physical design levels for a given technology.

In the logical design phase, detailed logic design is created and analyzed. In selected situations with today's technology, the logic of certain elements can be completely synthesized into technology specific gate level designs. In other cases, manual entry of the gate level logic of the design is required, at least in part. In any case, verification of the functionality and timing of the detailed level of the design is performed in this phase. Also in this phase, given high level layout of logic, *estimations* of testability, and certain design quality and reliability, can be performed (e.g., thermal analysis, power estimation).

In the detailed physical design phase, the detailed physical layout of the design object takes place. The detailed placement and wiring of the logic of the design is accomplished, guided by

the high level constraints and floorplanning steps from high level design, and detailed logic design. Next, an assessment of the impact of the placement and wiring on the overall timing of the design is performed (e.g., accurate timing analysis across design levels, hierarchical interconnect modeling, hierarchical parasitic extraction and modeling). Also in this phase, the *analysis and measurement* of design quality and reliability is performed (e.g., signal quality analysis, power grid analysis, thermal analysis, power analysis).

In the current environment there are a host of standard and defacto standards in use, including EDIF, CFI DR, PDEF, DEF, SPF, SDF, and several others. EDIF, CFI DR, and PDES (STEP AP210) are, to varying degrees, an attempt to address the needs of both the logical and the physical design data representation areas. However, there is no current adequate standard for logical connectivity that supports a file-based EDIF-like approach, as well as a programming interface DR-like approach, in a consistent way. In addition, there is no standard for chip physical design data, and the standard for MCM physical design data is not yet completed by EDIF. EDIF PCB/MCM was selected by the MCM ASEM alliance as the standard for MCM physical design data. CFI's current plans are to converge to a standard information model with EDIF (and eventually converge information models with PDES), so that the CFI programming interface evolves from a common information model.

5.4.2 Requirements

Below are listed the key information requirements needed to support the detailed logical and physical design phases for all package levels.

5.4.2.1 Standard Detailed Design Representation

A standard is required for an integrated representation (i.e., a file format and a programming interface based upon a common information model) for all detailed logical and physical design information and the interrelationships between them, including support for connectivity and related annotation of logical and physical design information. It must define a standard base from which to support all types of packages as defined in this document. It must include sufficient capability to support interoperability in a hierarchical and incremental design environment between all logical and physical design tools, as well as interoperability with the system design environment, and provide an effective interface for manufacturing build and mechanical design. It must support parameterized connectivity to enable a single primitive to support a variable number of inputs.

To adequately support detailed logical and physical design, there is a need for a design representation that maintains the relationships between the logical and physical components and access points and the logical connectivity model. This enables interoperability between tools that extract parasitics, calculate delays, and that do back annotation, manufacturing diagnostics and repair actions, etc.

In Table 5.1, "Impact of Design Size on Design Processing Times," on page 78, there are several major items of interest:

- The columns entitled "Transistors per Chip" and "ASIC Gates per Chip[1]" describe the projected number of transistors and chip area over the next decade
- The column entitled "Relative Design Data Size Increase" uses 1995 as a base for normal-

ization of data to show the *data explosion* for the years that follow:

- 1995 is “1”; i.e., the point of normalization.
- In 1998, the number of transistors nearly triples (from 5M to 14M). Therefore, whatever the design data size was in 1995, it will increase by a factor of 2.8 in 1998 (for ASICs), and so on. By the year 2010, the amount of design data will be 86 *times* the amount of data in 1995.

• The column entitled “Relative Simulation Times” uses a similar approach. Based on historical and empirical data, the amount of simulation time (e.g., CPU time) required to do simulation is n^2 times the number of circuits being simulated. Taking the design size information and squaring it yields the relative simulation time for that size of design (e.g., $2.8^2 = 7.8$). Again, by the year 2010, the projected simulation time for a design which is 86 times as large (for ASICs) as designs in 1995, will be 7396 *times* as long as it is in 1995. Simulation is only one of the design activities impacted by the tremendous growth in chip density; all design activities that tend to operate on “entire designs” will face this issue.

This data, based on the projections in the SIA Roadmap, clearly indicates the importance of the hierarchical design approach and incremental processing, as required approaches to support the very large chip designs of the future. We must take action *now* to support hierarchical and incremental design representations, and the design tools of the future must be designed to handle the processing of such design representations.

Table 5.1: Impact of Design Size on Design Processing Times

Year	Transistors per Chip	ASIC Gates per Chip	Relative Design Data Size Increase	Relative Simulation Times N^2
1995				
ASIC		5M	**1**	1
uP	12M		**1**	1
1998				
ASIC		14M	**2.8**	7.8
uP	28M		**2.3**	5.3
2001				
ASIC		26M	**5.2**	27
uP	64M		**5.3**	28
2004				
ASIC		50M	**10.0**	100
uP	150M		**12.5**	156

1. The National Technology Roadmap for Semiconductors, Semiconductor Industry Association, 1994, Overall Roadmap Technology Characteristics, Table 2 on page 16.

Table 5.1: Impact of Design Size on Design Processing Times

Year	Transistors per Chip	ASIC Gates per Chip	Relative Design Data Size Increase	Relative Simulation Times N^2
2007				
ASIC		210M	**42**	1764
uP	350M		**29**	841
2010				
ASIC		430M	**86**	7396
uP	800M		**67**	4489

This key standard must also support all requirements stated in 5.1 "Common Topics Across All Design Information", with special focus on 5.1.1 "Incremental Processing" and 5.1.2 "Hierarchical Processing".

5.4.2.2 Standard Detailed Design Representation - Extensions for PCB Packages

This package type has a number of unique requirements that must be addressed by any proposed standard for design and manufacturing release. The outline of a PCB is frequently odd shaped and designed in mechanical design. See 6 "Key Electronic Design and Test Interfaces" for key manufacturing and mechanical design interface requirements.

In certain instances, raw boards are designed and manufactured to allow choices of components in selected locations. In addition, this level of package can be reworked. Components may be added/deleted; wire connections can be added and deleted after initial build. These choices may have different lead spacings requiring a common bond site pattern. There is a need to be able to identify what has changed in a design. The standard must support requirements unique to this type of package.

5.4.2.3 Standard Detailed Design Representation - Extensions for MCM Packages

There are several types of MCMs. The MCM type that has chip(s) sitting in a well has unique requirements. The standard must support requirements that are unique to this type of package.

5.4.2.4 Standard Detailed Design Representation - Extensions for Chips, Macro-Cells

5.4.2.4.1 Concurrent Design Chip Requirements

To provide for EDA support for concurrent front end and back end physical design automation and aids, chip physical information needs to be integrated with the rest of the hierarchical electronics design representation. Support is needed for tracking of timing and performance, requirements, and design decomposition, across multiple levels of design and diverging design hierarchies. The tracking information is important in accurately back annotating, later more accurate design details back up to higher level, and earlier versions of the design, as part of the re-verification process.

The chip physical representation needs to support implementation of engineering change order (ECO), and other incremental change requirements by efficiently handling of redesign through re-use of previous physical design. The representation needs to support library changes, net list and other high level design representation changes, placement adjustments, and routing edits.

5.4.2.4.2 Logical to Physical Correlation Requirements

GDS-II Stream format is the current defacto standard for physical design maskout information, however, it does not support the correlation to the logical connectivity model. For extraction tools to have access to the rest of the design information needed to calculate delays and parasitic information, the extraction tools must either re-derive the original net and circuit relationships, or use the original design data that is not part of the interface data. It is more accurate and faster to store these relationships in the design data base than to re-derive the relationships.

5.4.2.4.3 Design and Tool Sharing Between Engineering and Manufacturing

Engineering tools used to view the physical design, are also required in manufacturing. Checks for manufacturability should be run earlier in the design process in the design shop. The engineering design tools use the relationships between the physical shapes and the logical connectivity. This engineering/manufacturing interoperability is another important reason to save these relationships in the physical design data, both in engineering and across the engineering/manufacturing interface.

5.4.2.4.4 Requirements Unique to Chip Packages

The physical design information for the chip (macro-cell and full chip), is much more voluminous than for other package types. There is no concept of unpopulated chips, or any concept of “assembly data” as in other package types. In an ASIC chip, the physical data from the circuit library is merged with all the other design elements or cells via placement into one collection. However, the design tools and analysis tools still require the correlation to the original logical design connectivity and schematic data.

5.4.2.4.5 Form of Incremental Release

The ASIC chip has a “front end of line” and “back end of line” concept. The back end of line data is on the last few mask levels in the manufacture of the chip. This consists of a restricted set of shape data from the circuit library and the interconnecting wire data. The standard must support requirements, such as this, that are unique to the chip level package.

5.4.2.5 Placement Data

There is a requirement for a standard to provide a means to efficiently record placement of cells, cores, and components to support a standard interface that will be used between:

- placement tools and wiring tools.
- placement and delay estimation
- placement and timing driven synthesis.

This information will be passed along with and be correlated to the logical connectivity

model, or may be passed as an incremental piece of information. Performance is a concern in tool interactions such as those listed above. In addition to the requirement for chip support, these concepts apply to higher level package types (i.e., boards, MCMs)

5.4.2.6 Standards to Support Floorplanning

There is a requirement to share information between the synthesis and initial floorplanning activities performed during system level design, and the detailed floorplanning with subsequent placement and wiring.

This data includes the specification of information described in 5.3.2.4 "Standards to Support Floorplanning", and additional refinements to meet detailed floorplanning requirements. This information must also be shared with the system level design phase so that high level floorplanning and synthesis can be effectively linked with detailed floorplanning, placement and wiring, etc.

5.4.2.7 Standard Language for Chip Layout Generators

Physical layout generators should use a language that is standard and universal. Just as most computers now have a language compiler that supports C for portability across operating system platforms, layout generators need a standard language to help with portability of the layout across different design groups and different toolsets.

The rationale for this requirement is to ensure that layouts can be re-used (or regenerated) by future layout programs as time elapses and design systems evolve. The longevity of IC layout designs can be better achieved through the use of a standard IC layout generation language. Pressures to achieve better design re-use, and portability of layouts, will only increase the desire to capture physical layout in standard tool-neutral formats.

5.4.2.8 Standards to Support Testability Analysis

5.4.2.8.1 Standards to Support Manufacturing Test Rules

There is a requirement for a standard method to describe the rules of a manufacturing test process, so that checking against those rules can be performed during the design process. These manufacturing process capability rules are required for all levels of packaging, to enable design for testability to be part of the design process.

Detailed testability analysis must be run early in the detailed design cycle, and as part of the final manufacturing test data generation. Subsets of testability analysis, are even run as part of the synthesis process. No matter when it is run, there is a need for the following:

- a manufacturing fault model (e.g., for traditional stuck fault coverage)
- delay fault model (for designs that are pushing technology performance).

These test rules are required to support testability analysis during the design process. When the design is pushing the technology limits for performance, then delay test generation, which requires a delay fault mode, is run.

Both of these fault models are addressed in 5.5 "Design and Technology Re-use".

5.4.2.8.2 Support for Test Vector Specification

In the process of performing testability analysis or final test data generation as described above, test vectors may be generated and captured. Any standard for test vectors must support the specification of this test vector information. This information is part of the key interface to manufacturing and is discussed further in 6.2 "Manufacturing Test Interface".

5.4.2.8.3 Standards to Support Component Self-Test

As circuit densities increase, use of test technologies that depend on inserted test logic (e.g., on-chip), will also increase. Automatic insertion of standard BIST and emerging test logic insertion must be supported in EDA Design tools.

5.4.2.8.4 Standards for Component Test Data Re-Use

As components and cores (subsets of chips), continue to be re-used in new designs, the test information for those reusable design objects must also be captured, so the test data can be re-used in the design and test of the new design. This also applies to components that are used in higher levels of package, so that the component test data can be used in testing the higher level package. This test information must be included in the appropriate re-use library.

The standards for test data re-use must address test for reusable chip components, including cores. Two different kinds of test information are required, including:

- Information to support test of the component (or core)

  This first requirement is satisfied by the appropriate conditions to enable controllability/ observability of the embedded component, so that existing patterns can be applied, using an in-circuit type of test strategy.

- Information to support test of the circuitry in which the component is embedded

  This second requirement calls for a standard that records modes of the component or core that allow a ‘flush-through” mode, so that patterns for the logic around the component can be flushed through the component when certain control conditions are met.

That is, this requirement documents *test-related* (as opposed to functional) behavior of the component. Armed with this kind of information an ASIC designer would know how to configure the component or core to propagate signals through the component, so that the surrounding circuitry may be tested.

5.4.2.9 Standards to Support Manufacturability Analysis

5.4.2.9.1 Standard Support for Manufacturing Build Rules

There is a requirement for a standard method for describing the rules of a manufacturing build process, so that design/manufacturing rules checking can be performed during the design process. These manufacturing process capability rules are required for all levels of packaging to enable design for manufacturability to be part of the design process.

5.4.2.9.2 Support for Virtual Manufacturing

Currently, there are key steps in the design of electronic packages to verify the functionality and timing of designs. This concept needs to be extended to include a virtual manufacturing process capability, in the EDA Design System, so that the manufacture of the device (i.e.,

macro-cell, chip, or board) can be simulated, as well as the functional operation of the device.

5.4.3 Recommendations

Action must be taken now to drive towards a converged and common core information model from which future standards effort(s) in design representation can be based. EDIF and CFI DR are currently working on a common core information model for logical connectivity. This work should be accelerated and be expanded if necessary to include any other applicable standards. Over time, other industry standards related to detailed design must be included in this convergence strategy.

The goal must be to develop a common information model from which a file format (e.g., extended EDIF), and a programming interface (e.g., extended CFI DR) can be developed. Once a common information model is agreed upon, all future releases of the standard for detailed design, should include a synchronized and simultaneous release of both the file format and the programming interface for the standard. This approach, will ensure that both the file format approach and the programming interface approach, are developed from a common information model base, and that either approach (or both) can be used to maximize data interoperability across and within EDA design systems.

The priority for this convergence should be for logical connectivity first (which applies to all levels of packaging), then support for detailed physical design. Efforts for physical design should be for board level packages first (based on EDIF work in this area), then physical design for chips. Every effort must be made to develop the physical design representations for PCB, MCMs, and chips in parallel, and as soon as possible.

5.4.4 Roadmap - Detailed Design

5.4.4.1 Converged Industry Standard for Logical Connectivity

The EDIF and CFI DR effort to develop a common core information model must be accelerated and completed as soon as possible. This information model primarily addresses logical connectivity. All detailed design industry standards that relate to logical connectivity must be part of this convergence effort, so that an industry-wide information model results from this work.

Based on the common core information model from the above effort, a new connectivity standard must be developed, that includes both a file format and a programming interface.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.2 Converged Industry Standard for Board Packages

Based on the above logical connectivity standard, extensions must be made to support the design and manufacturing build interface for high level board packages (i.e., PCA/PCB).

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.3 Converged Industry Standard for MCM Packages

Based on the above logical connectivity standard, extensions must be made to support the

design and manufacturing build interface for MCMs.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.4 Converged Industry Standard for Chip Packages

Based on the above standards, extensions must be made to support the design and manufacturing build for chip subsets (e.g., macrocells) and chips. This standard must include support for placement information as described in 5.4.2.4 "Standard Detailed Design Representation - Extensions for Chips, Macro-Cells".

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.5 Placement Data

Define a standard to efficiently record placement of cells, cores, and components, that will be passed along with and be correlated to the logical connectivity model, or may be passed as an incremental piece of information.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.6 Standards to Support Floorplanning

Define a standard to support floorplanning extensions required to support the interface between detailed floorplanning, placement, and wiring.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.7 Standard Language for Chip Layout Generators

Define standards for IC layout generation.

This item should be adopted as medium priority immediately, i.e., in the short (immediate) term.

5.4.4.8 Standards to Support Testability Analysis

5.4.4.8.1 Standards to Support Manufacturing Test Rules

Define a standard for Manufacturing Test Rules, to include traditional stuck fault models and delay fault models.

This item should be adopted as high priority immediately, i.e., in the near (immediate) term.

5.4.4.8.2 Support for Test Vector Specification

Define a standard for test vector specification.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.4.4.8.3 Standards to Support Component Self-Test

Define standards required to support insertion of BIST and other test logic.

This item should be adopted as medium priority immediately, i.e., in the near term.

5.4.4.8.4 Standards for Component Test Data Re-Use

Define standards required to support component test data re-use including:

- information to support test of the component (or core)
- information to support test of the circuitry in which the component is embedded.

This item should be adopted as medium priority immediately, i.e., in the short (immediate) term.

5.4.4.9 Standards to Support Manufacturability Analysis

5.4.4.9.1 Standard Support for Manufacturing Build Rules

Define a standard for specifying rules for the manufacturing build process, for all levels of packaging, to enable design for manufacturability.

This item should be adopted as high priority immediately, i.e., in the short (immediate) term.

5.5 Design and Technology Re-use

5.5.1 Environment

The National Technology Roadmap for Semiconductors states "new approaches must be found if industry is to continue on the 30% per-year, per-function cost reduction trend". Over the past few years, we've seen the development and exploitation of different technologies and methodologies to maintain this growth curve. One of these technologies includes using higher level design techniques, particularly languages such as VHDL and Verilog. Combined with design synthesis, this provided a great leap forward in representing design intent. Additional innovations in the very front end of design, however, are required and anticipated.

Incremental and hierarchical design techniques have also become increasingly important as we move up the complexity curve. Breaking a design up into manageable segments allows us to put more total engineering resources into the design, and exploit concurrent software and hardware (and eventually mechanical) co-design techniques.

Data management, in terms of information and complexity management, continues to grow in importance. No longer does a small group of engineers start and finish the entire design, self-contained, in their organizational cube. A single chip design today might be a massive system undertaking, with hundreds of engineers spread across global and corporate boundaries. Software management techniques for controlling the engineering data allows us to maintain our senses, without completely tripping over each other in the complex design.

Ever changing business practices in the '90s have enabled us to be more productive. We no longer try to do it ourselves. Who would have ever thought in the '80s you would see a joint chip designed by Apple and IBM? Or that TI and Hitachi could have ended up co-funding a new wafer fabrication facility for memory designs (Twinstar, Inc.). In today's business environment, we know we cannot do it ourselves. So, how can a design engineer take advantage of this global environment?

One answer is through design re-use.[1]

1. "Enabling Re-use Provides Product Design Productivity", George Chandler (Texas Instruments), Sumit Dasgupta (IBM/SEMATECH), Gary Panzer (Hughes Aircraft Company), 8/95.

Why Re-Use?

Re-use increases designer productivity. To begin with, there is less reinvention of the same thing. From a resource perspective, re-use allows incredible leveraging of other people's resources. Re-use

- enables others to independently develop building blocks in their area of expertise,
- enables you to leverage their work to produce new products using those pre-designed building blocks, and
- enables you to provide added value for your customers, at a reduced cost and with a shorter time to market.

By exploiting design re-use of existing or standard components, you can focus on more advanced technology and leading edge products for your customers, instead of spending time and money on something that has been done before.

Design Re-Use

Re-use has been used for years in other industries, from automotive to software. Once the Ford 302 cubic inch engine block was designed, it was used repeatedly in many different car styles for years to come. For years, re-use has existed at the component level; we already have an existing business model -- the standard component data books -- such as the TI TTL data book. Every engineer had one and used it to search for an appropriate component to use in his or her design. Re-use was easy. The function of each entity was generic, but limited in complexity.

Re-use did manage to work its way into the chip level design and founded a type of design methodology called “standard cells.” But there is still more to gain through wider usage of macrocells and functional blocks. Furthermore, there is still potential re-use at different levels of design and abstractions.

5.5.2 Requirements

Before identifying and reviewing the key requirements for design and technology re-use, some definitions are in order.

Levels of Re-use

For the purposes of this document there are several kinds of reusable design objects.

1) Reusable Design Object Specification

This specification is comprised of many *“datasheet” or design object specification* types of information about a reusable object. Examples of the information in these specifications are shown in the list below. This information base is the initial access point for the library of reusable design objects and is used to help designers determine if a reusable design object exists which can be reused in a new design situation. Typically, a Design Object Specification would exist if, and only if, one of the reusable function or component design objects also exists (see the definitions that follow below).

Only by knowing what the original designer intended will the person using the design object

(the “re-user”) have confidence in it. A certain level of knowledge is needed for confidence. The following list are the elements of knowledge we’ve found are most necessary. For each one, as it applies to each level of design (macro, chip, and system), the units may be different, but the basic concept applies.

- price
- time to market/cycle time
- performance
- function
- reliability/availability/serviceability/manufacturability
- power
- quantity
- technology
- physical design
- environment
- testability

For example, a different unit may apply for various uses of the knowledge element “price”. For a macro level, it may be simply the intellectual property price of using that element. For a system, it may be simply the unit price.

Every reusable design object (e.g., component, function, etc.) must have a Design Object Specification so the information can be accessed by appropriate search and retrieve methods.

2) Reusable Function

A function design object refers to the concept of a *simulatable design specification for the design object* that can be reused. The examples here might start with ALU chips or entire microprocessor designs for which a VHDL or Verilog simulation model is available. Several types of simulation models are in fact desired as are described below in 5.5.2.3 "Standards for Reusable Functions". This type of reusable design object is used to enable simulation of a candidate object in a new design context or to develop a new physical technology implementation of a previously designed function.

3) Reusable Component

A reusable component design object refers to the type of *component for which a physical implementation exists* that can be re-used. Examples are ALU chips, macrocells that can be embedded in larger chips, etc. These designs have previously completed physical implementations in given technologies that are largely predesigned from a physical design perspective.

Limited parameterization may be possible. There would typically exist technology libraries from one or more suppliers of such reusable design objects.

Standards Promote Re-use

Re-use can enable productivity across a wide range of the design cycle, from architecture design all the way to fabrication. However, each phase of the design cycle requires knowledge to be passed from the original designer to the “re-user”. This knowledge needs to be passed in a data format understandable to the creator and the re-user. Only through standards can this knowledge be confidently moved from user to user, through time, and across systems.

In the meantime, the current set of standards needs to be examined as a whole, not as individual parts. This set should be pruned where need be and grown/updated where holes exist so the entire set can become the preliminary information model eventually. This will also lead to the slight lagging of supporting library sets, and in the near-term, actual knowledge libraries.

Key Requirements

The following are the key requirements for design and technology re-use.

5.5.2.1 Standards for Reusable Design Object Classification Hierarchy

To make any standards reusable, a standard “dictionary of terms” or classification hierarchy must first be developed. Standard terms (reference the CFI EDB Data Dictionary and Pinnacles efforts) are necessary to enable search and retrieval engines to “find” the reusable objects in libraries. Each of the areas of requirements below also add to the requirement for a standard design object classification hierarchy, and to the information contained in the library for the objects that are present.

This item is high priority in the short term.

5.5.2.2 Standards for Reusable Design Object Specifications

In order for a design object to be re-used, there are key pieces of design intent (i.e., knowledge) as well as other information described in 1) Reusable Design Object Specification above. The minimum information required for design object re-use must be determined and a standard dictionary of terms and a specification of reusable design object content are required for this Reusable Design Object Specification.

This item must be addressed with high priority in the short term.

5.5.2.3 Standards for Reusable Functions

Simulatable HDL models must exist for a reusable function. Examples are VHDL, Verilog, and potentially other simulatable design descriptions. There are a host of different applications for such simulation models including architectural, performance, RTL level, and other simulatable models. Some or all of these models are required to enable effective re-use of the function. Standards are required in this area to define a standard dictionary of terms and a taxonomy to support the classification of reusable functions.

This item must be addressed with high priority in the short term.

5.5.2.4 Standards for Reusable Components

Technology implementations (i.e., detailed physical designs) of more complex design objects are rapidly emerging, and there are no standards for their representation. Examples, such as

macrocells (cores) and entire microprocessors, are candidates for reusable components. Standards are required to enable standardized access and re-use of such designs. Limited parameterization with some reusable components (e.g., bit width) is also a requirement.

This item must be addressed with high priority in the short (immediate) term.

5.5.2.5 Libraries of Reusable Design Objects

This requirement is to ensure the appropriate libraries of reusable components are put into place and they are based on the appropriate set of standards. The required libraries include the appropriate design object classification hierarchy and design object specification information described above, as well as entries in the appropriate function and component libraries.

This item must be addressed with high priority in the short (immediate) term.

5.5.3 Recommendations

The industry council should commission a task group as soon as possible to determine the base information for which standardization is required for *all* areas of re-use. This group should develop and expand upon this base work to enable the creation of standards for:

- Reusable Design Object Classification Hierarchy (Standardization of Dictionary)

  For this item, the CFI Electronic Data Book (EDB) and the PINNACLES work offers a base from which this definition of terms can evolve.

- Reusable Design Object Specifications

  In this area, the design object classification hierarchy defined above would be used to support the development of standards to support the interface to standards based search and retrieval engines which enable engineers to “find and obtain access to” reusable objects which meet their needs.

- Reusable Function

  Initially, reusable models in this category would be supported by the classification hierarchy and design object specification work above. Initial library members would consist of the existing and legacy functional simulation models described in VHDL and Verilog, etc. Over the long term, additional model types and classifications might well evolve to meet emerging requirements.

- Reusable Components.

  Finally, reusable physical component library members would consist of the existing legacy physical design information. Based on the design object classification hierarchy and design object specification efforts described above, additional physical design or component library types and classifications will evolve to meet technology requirements.

5.5.4 Roadmap - Design and Technology Re-Use

Re-use of information, models, and libraries is the key to increase designer productivity, leverage technology cheaply, and evolve an interchangeable set of standardized knowledge (knowledge libraries).

5.5.4.1 Standards for Reusable Design Object Classification Hierarchy

The industry council should commission a task group as soon as possible to determine the base information for which standardization is required. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy. Some progress has been made in this area in the RASSP program and that work should be considered by this task group. In addition, the work of the IEEE and Reuse Library Interoperability Group (RIG)[1] on software reusability should also be considered by this task group.

This task is high priority in the short term.

5.5.4.2 Standards for Reusable Design Object Specifications

The industry council should commission a task group as soon as possible to determine the base information required to support a standard interface for the “search and retrieval” of design objects. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy as described above.

This task is high priority in the short term.

5.5.4.3 Standards for Reusable Functions

In this category, the first order of business must focus on the various simulation requirements to support reusable functions. As documented in 5.3 "System Level Design", high level simulation models at various levels are extremely important during the early phases of design, where hardware/software trade-off analysis and overall system performance are evaluated. Standards for creating such reusable simulation models are needed for:

- Architecture Design
- Performance Modeling
- Algorithmic Design
- Functional Design

Again, the industry council should commission a task group as soon as possible to determine the base information required to support a standard taxonomy and dictionary of terms in this area to provide a base to develop and classify reusable functions. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy as described above. The work done in the RASSP program to define a taxonomy for simulation models should be considered by this task group.

This item is high priority in the short term.

5.5.4.4 Standards for Reusable Components (Technology Implementations)

In the area of detailed design, both logical and physical design standards are required to support the concept of reusable components. Again, as documented in 5.4 "Detailed Design", detailed logical and physical information in standard form is required to enable reuse at the component (physical) level.

1. Standard Reuse LIbrary Basis Data Interoperability Model (BIDM), RIG Proposed Standard RPS-001 (1993), Approved 4/1/93, Revised 1/3/95.

Technologies which must be supported in these libraries include:

- Discrete Components (e.g. transistors, resistors, etc)
- Integrated Circuits
  - Custom ICs
  - DSPs
  - ASIC Cells
  - ASIC Cores
  - FPGA
  - PLD/CPLD
  - Memory (RAM, SRAM, DRAM)
- Assemblies
  - PCB (Digital, Analog, Mixed Signal, RF, Multilayer, Flexible)
  - MCM (MCM-Laminated/Deposited/Ceramic (Digital, Analog, MIxed Signal, RF...)
- System

Information in the detailed design category for reusable component libraries must include:

- Technology Mapping (to logic design)
- Logic Verification & Analysis
- Electrical Simulation & Analysis
- Test
- Physical Design
- Verification/Rule Checking
- Parasitic Analysis/Simulation

Also in the category of reusable component information is the specific information needed to support the release of the necessary manufacturing build information, including:

- Design to Manufacturing
- Fabrication
- Configuration Management

Refer to 6.1 "Manufacturing Build Interface" for additional information.

Similarly, in this category, the industry council should commission a task group as soon as possible to determine the base information required to support a standard taxonomy and dictionary of terms to provide a base to develop and classify reusable components. The categories of work which follow depend heavily upon the existence of a standard dictionary of terms, and a classification hierarchy as described above.

This item is high priority in the short term.

5.5.4.5 Libraries of Reusable Design Objects

Each reusable design object (e.g., reusable function or component) must have a Design Object Specification. This information is required so that the information which makes those objects easily found by search and retrieval engines which are compliant to the standard Design Object Classification Hierarchy. This design object specification must include *many* “datasheet” or design object specification types of information such those listed in 5.5.2 "Requirements"above.

Libraries of reusable design objects of the types described in this section must be built to the standards described in this section to make them accessible by compliant search and retrieval mechanisms. In order to capitalize on the significant productivity increases offered by design reuse, we must be able to “find and get access to” reusable design elements.

We must also be able to co-exist with the legacy libraries and their contents while we migrate to a longer term more “reusable” library strategy as described in this section. The strategies described in 2.1.4 "Design Information Roadmaps" and in Figure 4.2— "Open EDA Data Interoperability Architecture" overviews a strategy for how that can be enabled.

This task is high priority in the short term.

6 Key Electronic Design and Test Interfaces

6.1 Manufacturing Build Interface

6.1.1 Current Environment

At various points in time, there is a need for information exchange and sharing between the development and manufacturing organizations. While this phase is primarily characterized by the “release” of the final design of the design object to a manufacturing organization for the purpose of performing the manufacture of the product, it is also important for manufacturing to have an early involvement in the design process to facilitate effective manufacturability of the product. Through “early manufacturing involvement (EMI)”, design for manufacture can be part of the process at each of the design phases.

The increasing use of re-usable chip subsets (cores) present a new test challenge. Designers are often forced to tie the core interface to I/O pads in order to gain access both to the core itself and to the circuitry in which the core is embedded. This design technique is unsatisfactory for sub-micron design features. DFT techniques will have to be applied to cores (see 6.2 "Manufacturing Test Interface" for additional information), with corresponding documentation requirements.

6.1.2 Requirements

Manufacture (and test) engineers need information on product design, tooling. tolerances, fabrication sequences, dimensional requirements for design features, bare die and bare board testing information. Board assemblers need information on components, point of origin, assembly sequences, board aide relationships, location of fiducials, and electrical test vector information for in-circuit or functional testing. This information needs to be captured during the design process, even though much of this information is manufacturing build or test specific. The goal of the manufacture and/or test engineer is to ensure that the information they need to complete their job is provided in a timely and efficient manner.

6.1.2.1 Standard Support for Manufacturing Build Specification

There is a requirement for standard manufacturing build specifications for all levels of packaging including manufacturable chip subsets (macrocells) and entire chips, MCMs and boards (PCA/PCB). A standard interface for passing all manufacturing data that is available from the design process into manufacturing is required (note that not all the data that manufacturing needs is available during the design phase).

This interface must include the support for logical to physical relationships discussed in 5.4 "Detailed Design". Refer to that section for additional information on key manufacturing build interfaces such as

5.4.2.2 "Standard Detailed Design Representation - Extensions for PCB Packages"

5.4.2.3 "Standard Detailed Design Representation - Extensions for MCM Packages"

5.4.2.4 "Standard Detailed Design Representation - Extensions for Chips, Macro-Cells"

6.1.3 Recommendations.

Action must be taken now to drive towards a converged and common core information model from which future manufacturing interface standards effort(s) in design build representation can be based.

The work described in 5.4 "Detailed Design" should therefore be accelerated.

Over time, other industry standards related to the design-manufacturing interface must be included in this convergence strategy. Every effort must be made to develop the physical design representations suitable for manufacturing of PCB, MCMs, and chips in parallel, and as soon as possible.

These design and manufacturing interface standards should be integrated and extended to provide complete coverage of all key manufacturing interface requirements. Note that we may end up with multiple standards (use models built from a common information model) addressing the same kind of information but for different levels of package.

The standard manufacturing build interface requirements must be a convergence of standards that are currently under development (including EDIF PCM/MCM, CFI DR, and STEP AP2xx). We should track these standards and ensure that all key requirements are satisfied by the standards. This provides an opportunity to develop a single set of standards for the “manufacturing-to-design” information flow.

6.1.4 Roadmap - Manufacturing Build Interface

6.1.4.1 Standard Manufacturing Build Interface for Chips (and Macrocells)

Refer to section 5.4.2.4 "Standard Detailed Design Representation - Extensions for Chips, Macro-Cells" for details on this standard.

This item should be addressed as high priority in the short term.

6.1.4.2 Standard Manufacturing Build Interface for PCBs

Refer to section 5.4.2.2 "Standard Detailed Design Representation - Extensions for PCB Packages" for details on this standard.

This item should be addressed as high priority in the short term.

6.1.4.3 Standard Manufacturing Build Interface for MCMs

Refer to section 5.4.2.3 "Standard Detailed Design Representation - Extensions for MCM Packages" for details on this standard.

This item should be addressed as high priority in the short term.

6.2 Manufacturing Test Interface

6.2.1 Current Environment

Design for test is frequently addressed as a phase late in the cycle after the architecture, high level and detailed level design has been done (e.g., design for testability, fault modeling/analysis and grading, ATPG techniques, insertion of low-overhead BIST). A key to success in the designs of tomorrow will be to consider design-for-test as early as possible in the design cycle. In addition, evolving test methods such as self-test must be addressed.

A major impediment to new design-for-test processes is the lack of any standard representation for test information. Designers are unable to express test strategies or configurations in a way that can be understood by many tools. Designers are therefore limited to single-tool solutions for test. The lack of a fully-developed and elaborated test standard further increases product cost and development time when outside components, including cores, bare die, and packaged ICs, are inserted into a design. If test information, especially design-for-test information, can accompany these components, then tests for the new design can be developed more efficiently. However, the transmission of test information for these components will not be successful in the absence of test standards.

As discussed in 3.1 "Emerging Paradigm Shifts", industry is moving to integrate product development processes. An example of this which is directly related to the test-related functions of CAD systems is the test coverage management practice being developed on the RASSP program. Better integration of design and test information is important if the design and test processes are to be concurrent.

6.2.2 Requirements

Manufacturing test engineers have a need for much of the same information as was required for manufacturing build. In addition, there are some specific requirements such as those listed below. Refer to 5.2.2 "Simulation and Test Control" for additional information related to these requirements.

In addition, many of the requirements of manufacturing test are being driven earlier into the development process. Refer also to 5.4.2.8 "Standards to Support Testability Analysis" for test related requirements on the design process.

6.2.2.1 Standards for Test Vector Specification

There is a requirement for a standard form(s) of test vector specification which can be used on any manufacturing tester which is compliant to the standard. The specific requirement is to define how digital test vectors can be mapped from one format to another.

There is a large number of digital test vector standards, both formal and de facto. Formal standards include WAVES (IEEE Standard 1029.1), DTIF (IEEE P1029.4), and the new DASC proposal. De facto standards include those owned by Summit Design and by Teradyne. Many companies have well-established internal formats.

The plethora of formats impedes communication between tools and the creation of new, specialized tools. A single format would solve this problem, but, any attempt in the short term to

impose a single test vector standard is probably doomed to failure. These different formats each solve a problem for a particular constituency, and the needs of those constituencies must be respected.

6.2.3 Recommendations

We should start by harmonizing standards for digital test vectors. There are several formal and de-facto standards currently existing, most of which are quite capable. A new standard is not needed in the short term, but the existing standards may need improvement. We strongly recommend that test vector formats be converged through a common, core information model. Convergence establishes a formal relationship between multiple standards without forcing any standard to actually change. Over time, based on a common core information model for test information, the potential for a new standard in this area may gradually increase.

We must also work with developing standards for expressing test information that is more complex than digital test vectors. Some work is already being done in the IEEE. Here again, we have an opportunity to encourage a single set of standards with no overlap.

An important missing piece is a standard library of test methods that can be re-used among designs. This is similar in value to reusable hardware components. Such a library, supporting the Roadmap standards for test, will jump-start acceptance of the Roadmap standards.

To address requirements for component test data re-use, we can start by looking at existing, commercial formats that deal with in-circuit test. In an in-circuit test, the pins of components that are not being tested are placed in a high impedance state so as not to interfere with the test signals being applied to the component under test.

Refer to 5.4.2.8 "Standards to Support Testability Analysis" for additional discussion on these topics.

6.2.4 Roadmap - Manufacturing Test Interface

6.3 Mechanical Design Interface

6.3.1 Current Environment

The interface between mechanical design and electronic design for higher level packages such as boards (including MCMs, PCAs, PCBs) is a very important interface.

6.3.2 Requirements

6.3.2.1 Support for Electronic Design and Mechanical Design Interface

There is a requirement for a standard to support the specification and annotation of selected board level mechanical data and electrical design data in both the electronic design and mechanical design databases. Both design disciplines need to know the package outline, the precise location of connector data, and information about mechanical holes and restricted areas, cable sockets, and SMA plugs. The electrical domain needs to understand the relation-

ship between the precise location of an instance of a component part number and the electrical connectivity and physical design data required to implement the connection.

6.3.3 Recommendations

6.3.4 Roadmap - Mechanical Design Interface

6.4 Software Design Interface

6.4.1 Current Environment

The interface between software design and electronic design is becoming much more important than ever before, given the cost of designing hardware (due to the size of designs) and of software (since most complex electronic systems are software driven). Many of the key innovations in the design of complex systems are centered around the interface of software and hardware components. For architectural and high level design, evaluation of early hardware designs using techniques such as performance modeling is an emerging technique. Similarly, virtual prototyping enables designers to maintain a consistent view of the hardware developed for the software (and vice-versa) throughout the design cycle, thus easing the integration task and reducing the frequency and cost of redesign. Co-simulation of the detailed hardware and software design is used to verify the correct system behavior before the hardware is fabricated.

Virtual prototyping is defined to be a comprehensive model of a system or component that models the external and internal temporal, data-value, functional, and structural combination of aspects for the system or component. A virtual prototype may be written at any level of abstraction. However, the most significant usage of the term, virtual prototypes, occurs at and below the network architectural level.

In practice, virtual prototyping is done to allow the hardware and software designers to maintain a consistent view of the system as they develop their respective portions of the design. The hardware designers see an updated representation of the software and its requirements as they work and the software designers have an up-to-date representation of the hardware and its capabilities to target their software. This greatly aids the integration task and reduces redesign.

Hardware/software co-design is the simultaneous consideration of hardware and software within a system design and includes the co-development and co-verification of hardware and software in the system (from Myers, Bard, and Schaming - HW/SW Co-design white paper, Oct 94). Currently, co-simulation is used to ensure the designed system operates as intended, typically with a fully-detailed hardware and software model. This is a low level of abstraction.

Co-specification entails deriving specifications for hardware and software components of a system based on the requirements and system specification. Co-specification is currently not typically performed in a formal sense, but rather glossed over. This is at a high level of abstraction.

6.4.2 Requirements

The importance of software within the system design task becomes more important over time as system developers demand increasing performance and flexibility. The need for increased automation to aid in the development of hardware, software, and the interface between hardware and software is acute, as software development is often the bottleneck to system design, thus dictating the schedule and cost of the overall system design.

Increasing size and complexity of designs implies the importance and difficulty of hardware/software co-design will continue to grow. The following requirements need to be addressed:

6.4.2.1 Standards for Simulatable Specifications

A standard to define system requirements and behavior via simulatable specifications is needed. Simulatable specifications should support hierarchical top-down design, include system behavior requirements, design constraints such as size, power, and weight, and tests to ensure the conformance of a design to the requirements. The semantics of simulatable specifications must be precisely defined. Simulatable specifications must be flexible enough however, to support different design methodologies and implementations.

6.4.2.2 Standards for Hardware/Software Partitioning

The ways to partition tasks into hardware and software and to map software tasks to processors need to be addressed via recommended practices, standard heuristics (akin to using simulated annealing as a standard approach to placement and routing), or standard algorithms.

How to determine a set of viable alternative design approaches based on a mix of hardware and software components is a key step in hardware/software co-design that must be better understood.

[Editor’s note: this requirement is unclear as of this writing....because the discussion appears to be more functional in nature (i.e., “standard practices/heuristic/algorithms), and thus may not be appropriate for this document on EDA industry standards. This document is not designed to address algorithms, but to focus on standardization to support EDA tool interoperability, plug and play, etc. For the moment, this paragraph is left here to make sure something important is not lost]

6.4.2.3 Standard Support for Rapid Hardware/Software Design Evaluation

Standard libraries or practices for performance evaluation via simulation (as with VHDL), numerically, or analytically is needed to support rapid searching through the hardware/software design space.

6.4.2.4 Standard Interfaces for Modeling Hardware/Software

A standard interface or set of interfaces is needed to model the hardware as viewed by the software and the software as viewed by the hardware.

6.4.2.5 Standards to Support Virtual Prototyping

Standard practices for virtual prototyping are needed which support different levels of design detail.

6.4.2.6 Standards to Support Co-Simulation

Standards for co-simulation and testing are necessary to provide the capability of executing software on simulated hardware. Testing the hardware and software with a standard hardware/ software equivalent of “test vectors” as in IEEE WAVES would help facilitate tool portability and interoperability.

6.4.3 Recommendations

In view of the requirements stated above, the recommendations in this area include:

- Develop a taxonomy of terms and definitions, using the RASSP taxonomy as a starting point, perhaps including the classification of types of systems to focus on in the domain of standards

  - real time, embedded, general purpose
  - shared, dedicated; dynamic or static scheduling
  - serial, parallel

- Develop standard definition, requirements, and/or form of simulatable specification
- Develop partitioning techniques, algorithms, heuristics
- Develop modeling standards/techniques/libraries
- Develop standard hardware/software interface models
- Develop standards for virtual prototyping with consistent hardware/software views through various levels of design detail
- Develop standard means to verify correct behavior of hardware and software together

6.4.4 Roadmap - Software Design Interface

Standards and standard practices need to be developed for many of these areas. Some products and university research tools exist that could become defacto standards, although it is probably too early to consider them mature enough to be included as standards.

6.4.4.1 Focus on Standardizing the Taxonomy of Modeling

Appendix A - The Roadmap Working Groups

The following people from across the EDA industry and around the world participated in the development of the EDA Industry Standards Roadmap. The three working groups are shown with the working group chair(s), and working group champions or other key participants are identified within each group.

- EII - See Table A.1: "EDA System Interoperability and Integration Working Group (EII)"
- DDM - See Table A.2: "Design and Data Management Working Group (DDM)"
- TLM - See Table A.3: "Technology Libraries and Models Working Group (TLM)"

Table A.1: EDA System Interoperability and Integration Working Group (EII)

Name	Representing	Background	Participation
Grant Martin	Cadence	User	Co-Chair
John Teets	CFI	CAD Integrator	Co-Chair & Champion Design System
Elfriede Abel	ECSI Europe	Standards	Review
Malcolm Ash	Mentor Graphics	Developer	Review
Dieter Bergman	IPC	Standards	Review
Victor Berman	Cadence	Developer	Review
Dick Bushroe	HP/SEMATECH	CAD Integrator	Roadmap Sponsor
George Chandler	Texas Instruments	User	Co-Champion Reuse
Ron Christopher	Independent	CAD Integrator	Champion Design Information
Jim Clark	Electronic Tools Company	Developer	Review
Don Cottrell	CFI	Standards	Review
Sumit Dasgupta	IBM/SEMATECH	Developer	Assist Champions
Shaun Devlin	Ford	Standards	Review
Mike DonTigny	Mentor Graphics	Developer	Revierw
Peter Eirich	Westinghouse	Standards	Review
John Eurich	Engineering DataXpress	Developer	Review
Mark Falco	Lockheed Martin	CAD Integrator	Review
Gary Ferrari	TechCircuits	User	Review
Rob Fletcher	IEEE	Standards	Review
Steve Fortier	Intermetrics, Inc	Government	Champion Key Interfaces
Donna Fritz	EDAC	Standards	Review

Table A.1: EDA System Interoperability and Integration Working Group (EII)

Name	Representing	Background	Participation
Anthony Gadient	PDES, Inc.	User	Review
Rita Glover	EDA Today	EDA Consultant	Review
Steve Grout	SEMATECH	User	Review
Tamotsu Hiwatashi	Toshiba	CAD Integrator	Review
Jan Johansson	Ericsson	User	Review Only
Hilary Kahn	University of Manchester	Standards	Review
B. J. Kalathil	Lockheed Martin ATL	CAD Integrator	Review
Bipin Chadha	Lockheed Martin ATL	CAD Integrator	Review
Jan-Olof Kismalm	Ericsson	User	Review Only
Eskil Kjelkerud	Ericsson	User	Review Only
Mike Mcllrath	MIT		Review
Larry Melling	IKOS	User	Review
John Mermet	ECSI Director	Standards	Review
Joe Morrison	Loral	User	Review
John Murphy	Cadence	Developer	Review
Mike Krause	Nortel	User	Review
Mika Nuotio	Ericsson	Developer	Review
Rick Ong	Motorola (RASSP)	CAD Integrator	Review
Gary Panzer	Hughes Aircraft/RASSP	User	Co-Champion Reuse
Greg Peterson	Wright Patterson AFB	Government	Software I/F
Sishpal Rawat	Intel	User	Review
Patty Rusher	EIA/EDIF	Standards	Review
Mike Tong	AT & T	CAD Integrator	Review
John Welsh	Lockheed Martin ATL	CAD Integrator	Review
Tom Vanderberge	Texas Instruments	User	Review

Table A.2: Design and Data Management Working Group (DDM)

Name	Representing	Background	Participation
Don Cottrell	CFI	Developer	Co-Chair
Danny Davis			Review Only
Dieter Bergman	IPC	Standards	Review
Ron Christopher	Independent	CAD Integrator	
Robert Fletcher	IEEE	Standards	
Ted Frederick	Cadence	Developer	Co-Chair
A. J. Incorvaia	ViewLogic	Developer	
Donna Fritz	EDAC		
Rich Goldman	EDAC, Synopsis	Developer	Review
Steve Grout	SEMATECH	User	
Jan Johansson	Ericsson	User	Review Only
Sunil Joshi	Sun	User	Review Only
B. J. Kalathil	Lockheed Martin ATL	CAD Integrator	
Jan-Olof Kismalm	Ericsson	User	Review Only
Eskil Kjelkerud	Ericsson	User	Review Only
John McClintock	Mentor Graphics	Developer	
Mika Nuotio	Ericsson	Developer	
Patty Rusher	EIA/EDIF	Standards	
Lutz Treutler	FED		Review Only
Danny Davis	Intermetrics		
Tom Vandenberge	Texas Instruments	User	Review
Nikolay Vitsyn	Independent		
Khan Vu	Sun	CAD Integrator	
John Welsh	Lockheed Martin ATL	CAD Integrator	
Jim Wilmore	HP	CAD Integrator	

Table A.3: Technology Libraries and Models Working Group (TLM)

Name	Representing	Background	Participation
Ron Waxman	Independent Consultant	Standards	Co-Chair
Steve Schulz	Texas Instruments	User	Vice-Chair
Dieter Bergman	IPC	Standards	Review
Dennis Brophy	Mentor Graphics	Developer	Review
Ranjit Chandra	Cadence	Developer	Review
Shirshen Chang	Synopsys	Developer	Review
Arindam Chatterjee	Texas Instruments	User	Review
Ron Christopher	Independent	CAD Integrator	Review
Don Cottrell	CFI	Standards	Review
Mike Emley	ViewLogic	Developer	Review
John Evans	Lockheed Martin	CAD Integrator	Review
Shahram FamorZadeh	Georgia Tech		
Elisa Finnie	Independent	Developer	Review
Rob Fletcher	IEEE	Standards	Review
Donna Fritz	EDAC		
Ian Getreu	Analogy	Developer	Review
Rich Goldman	EDAC, Synopsis	Developer	Review
Steve Grout	SEMATECH	User	Review
Carl Hein	Lockheed Martin	CAD Integrator	Review
Mitch Heins	CFI	CAD Integrator	Review
Shanker Hemmady			
Jan Johansson	Ericsson	User	Review Only
B. J. Kalathil	Lockheed Martin ATL	CAD Integrator	Review Only
Jan-Olof Kismalm	Ericsson	User	Review Only
Eskil Kjelkerud	Ericsson	User	Review Only
Mike Krause	Nortel	User	Review

Table A.3: Technology Libraries and Models Working Group (TLM)

Name	Representing	Background	Participation
Pat McHug	Army Research Labs	Government	Review
Mika Nuotio	Ericsson	Developer	Review
Hank Oredson	Mentor Graphics	Developer	Review
Harry Parkinson	DEC	User	Review
Susan Runowicz-Smith	LSI Logic	User	Review
Patty Rusher	EIA/EDIF	Standards	Review
William R. Simpson	Institute for Defense Analysis	User	Review
John Teets	CFI	Standards	Review
Yatin Trevedi			Review
Tom Vandenberge	Texas Instruments	User	Review
Steve Waterbury	NASA	User	Review
John Welsh	Lockheed Martin ATL	CAD Integrator	Review
Hitoshi Yoshizawa	NEC	User	Review

Standards Survey

The IEC would like to offer you the best quality standards possible. To make sure that we continue to meet your needs, your feedback is essential. Would you please take a minute to answer the questions overleaf and fax them to us at +41 22 919 03 00 or mail them to the address below. Thank you!

Customer Service Centre (CSC)

International Electrotechnical Commission
3, rue de Varembé
1211 Genève 20
Switzerland

or

Fax to: **IEC**/CSC at +41 22 919 03 00

Thank you for your contribution to the standards-making process.

Nicht frankieren
Ne pas affranchir

Non affrancare
No stamp required

A Prioritaire

RÉPONSE PAYÉE

SUISSE

Customer Service Centre (CSC)
International Electrotechnical Commission
3, rue de Varembé
1211 GENEVA 20
Switzerland

Q1 Please report on **ONE STANDARD** and **ONE STANDARD ONLY**. Enter the exact number of the standard: *(e.g. 60601-1-1)*

.....

Q2 Please tell us in what capacity(ies) you bought the standard *(tick all that apply).* I am the/a:

purchasing agent ❑
librarian ❑
researcher ❑
design engineer ❑
safety engineer ❑
testing engineer ❑
marketing specialist ❑
other.....

Q3 I work for/in/as a: *(tick all that apply)*

manufacturing ❑
consultant ❑
government ❑
test/certification facility ❑
public utility ❑
education ❑
military ❑
other.....

Q4 This standard will be used for: *(tick all that apply)*

general reference ❑
product research ❑
product design/development ❑
specifications ❑
tenders ❑
quality assessment ❑
certification ❑
technical documentation ❑
thesis ❑
manufacturing ❑
other.....

Q5 This standard meets my needs: *(tick one)*

not at all ❑
nearly ❑
fairly well ❑
exactly ❑

Q6 If you ticked NOT AT ALL in Question 5 the reason is: *(tick all that apply)*

standard is out of date ❑
standard is incomplete ❑
standard is too academic ❑
standard is too superficial ❑
title is misleading ❑
I made the wrong choice ❑
other .....

Q7 Please assess the standard in the following categories, using the numbers:
(1) unacceptable,
(2) below average,
(3) average,
(4) above average,
(5) exceptional,
(6) not applicable

timeliness.....
quality of writing.....
technical contents.....
logic of arrangement of contents .....
tables, charts, graphs, figures.....
other .....

Q8 I read/use the: *(tick one)*

French text only ❑
English text only ❑
both English and French texts ❑

Q9 Please share any comment on any aspect of the IEC that you would like us to know:

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

ISBN 2-8318-5762-7

ICS 35.240.50

Typeset and printed by the IEC Central Office
GENEVA, SWITZERLAND