A TUTORIAL INTRODUCTION TO THE ELECTRONIC DESIGN INTERCHANGE FORMAT

John P. Eurich

Independent Consultant
2856 Sycamore Way
Santa Clara, CA 95051
(408) 243-8786

Abstract

The Electronic Design Interchange Format (EDIF) is a neutral data format, through which electronic design information can be exchanged between CAD/CAE systems, semiconductor foundries, fabrication and assembly facilities, etc. The EDIF standard has been under development since November 1983. This paper discusses the history of its development, the organization which supports it, the goals and guidelines followed by the professionals who are developing the standard, and numerous details of the standard itself. Also, an analysis of how well the standard currently covers today's electronic design information is presented, as well as a methodology for developing translators which read and write the format.

Introduction

The Electronic Design Interchange Format (EDIF) is a neutral data format, through which electronic design information can be exchanged between CAD/CAE systems, semiconductor foundries, fabrication and assembly facilities, etc. Although today the EDIF standard is by no means complete, it is solid enough in certain areas to support the interchange of numerous types of design data. Many companies and universities have developed EDIF translators and many products have been announced which support EDIF.

The purpose of this paper is to present the status of the Electronic Design Interchange Format (EDIF) Standard as it is defined as of this writing (March 1986).

The EDIF, by its very nature, is a continuously evolving standard, and thus is currently being modified and enhanced. Therefore, this paper will address the latest formal specification [1], version 1 1 0, and any known proposed enhancements which are close to being included in the standard [2]. Currently the EDIF standard is being modified to improve its ability to handle Electronic Schematics, Netlists, and Printed Circuit Board information.

History and Organizational Structure

The reason the EDIF effort was started has been explained in many other documents [1, 3, 4, 5]. Briefly, the effort was started by a consortium of workstation and semiconductor vendors, consisting of: Daisy Systems Corp., Mentor Graphics Corp., Motorola, Inc., National Semiconductor Corp., Tektronix, Inc., and Texas Instruments, Inc. Each had been experimenting with its own data interchange formats, and thus saw the need for a neutral data format through which electronic design data could be exchanged between companies and CAE/CAD systems. The original effort started in November 1983. The first public specification, a preliminary draft [3], was released in May 1984, followed by a formal specification release in March 1985 [4]. The latest release [1], version 1 1 0, was in November 1985. The next release is scheduled for the summer of 1986.

The EDIF standard is evolving through the efforts of people who work as members of a number of committees: the Steering Committee (SC), the Technical Committee (TC), numerous Technical Subcommittees (TSC), and a Public Relations Committee (PRC). The SC is responsible for the administrative activity of the EDIF, and for the release of each new version of the standard. The TC is responsible for the technical content of the EDIF specification, for incorporating reviewer and TSC feedback into the specification, for maintaining consistency across all areas of the specification, and for coordinating and monitoring the efforts of the TSCs. The TSCs are made up of experts in specific fields and are responsible for developing proposals for enhancements which address specific areas of electronic design. The current active TSCs are: the Test TSC, the Behavioral TSC, the Procedural TSC, the Physical Design Rule TSC, and the Printed Circuit Board TSC.

Besides committees, professionals can get involved with the EDIF effort through EDIF Users' Groups. Currently there are active Users' Groups in the United States, United Kingdom, and West Germany. For more information on EDIF Users' Groups, EDIF Committees, and copies of the latest EDIF Specification, contact:

The EDIF Users' Group
Design Automation Department
Texas Instruments
P.O. Box 225474, MS3668
Dallas, Texas, 75265

0738-100X/86/0000/0327$01.00 ©1986 IEEE

Development Work Flow

Typically the development of a new EDIF Specification starts with the Steering Committee setting the requirements for the next release. The Technical Committee relays these requirements on to the Technical Subcommittees. If a new TSC is required, then the TC will search for a TSC Chairperson, who will form the new TSC. Feedback from users is sent to EDIF committees, via the EDIF Users' Groups.

Work progresses by the TC and the TSCs through assignments, meetings, and the use of computer resources (donated by Octal Computer of Mountain View, CA), and electronic mail. The TC reviews proposals from the TSCs and sends comments back to the TSCs asking for changes or enhancements to the proposals. The TC and the TSCs continue to interact this way, continually resolving problems.

Eventually the TC merges the new proposals and Users' Group feedback into a new draft of the specification. This draft is reviewed by the SC, TC, TSCs, and others. The review and modification process continues until the SC feels that the draft specification is solid enough for formal release. A release version is then created and printed. Those on the EDIF mailing list are notified of the new version's availability.

Goals and Guidelines

The EDIF Committees have been guided by a set of goals and guidelines, such as:

1. The EDIF will become a standard by use, not edict. This implies that it be released for testing regularly, and that feedback must be incorporated into future releases.
2. The Standard will be as neutral as possible, so that it can represent information from numerous systems.
3. The EDIF Standard will be in the Public Domain.
4. The initial Standard will contain a framework for future work.
5. Later versions will focus on improvements to specific information areas (e.g. Schematics, Printed Circuit Board, etc.).
6. The syntax will be consistent throughout.
7. The syntax will be extensible, so that the format can keep up with changing technology.
8. The syntax will be simple, so that implementations of translators can be made easily and quickly.
9. The Standard will support a mechanism which will allow it to be upward compatible.
10. The Standard will support an unambiguous semantics and syntax.

Syntax and Semantics

The Basic Syntax of EDIF

One of the main goals of the EDIF effort is to keep the syntax simple so that translators can be easily and quickly implemented. Another goal is to have a syntax which is easily extended so that the standard can keep up with evolving technology. It is because of these goals that a "lisp like" syntax was chosen by the TC for the EDIF standard.

The basic form of the EDIF syntax is the "EDIF Statement" [1]:

(*keywordName* { *form* })

a left parenthesis followed by a legal and well defined EDIF keyword name, followed by "forms" (a well defined sequence of "symbolic constants", "identifiers", "primitive data", or EDIF statements), followed by a right parenthesis.

More formally the syntax of the "form" is:

SymbolicConstant | *identifier* |
primitiveData | *EDIFStatement*

A SymbolicConstant is a well defined EDIF name, an identifier is a legal sequence of characters representing the name of a translated object or data group, and primitiveData is a number, string, or boolean.

The simplicity of this syntax enables an "EDIF Reader", a translator that reads EDIF syntax, to easily finds its way through an EDIF file by keying off left parentheses. After each left parenthesis is a legal keyword, which unambiguously defines the syntax of the form which follows. If the keyword represents data that the Reader wishes to ignore, then it only needs to skip data until it finds the next matching right parenthesis. If the Reader comes across a keyword that it does not recognize (possibly because the Reader is not current with the latest EDIF version), then it only needs to issue a warning message, and skip to the next matching right parenthesis to recover.

The EDIF Standard is easily extended by carefully adding new EDIF Statements to the syntax. If the new EDIF Statements are only added as optional constructs, then another goal of the EDIF Standard, upward compatibility, is attained.

Separate Name Spaces

Another goal of the Standard is to eliminate any naming clashes between EDIF Keyword names, EDIF Symbolic Constant names, and Identifier names. In the EDIF Specification this is referred to as maintaining "separate name spaces". This was required because the TC did not want to place too heavy a burden on the "EDIF Writers", a translator which creates EDIF files, and which must assign EDIF identifiers to objects and data blocks.

This was easily attained for Keyword names since Keywords are the only type of name that can immediately follow a left parenthesis. Clashes between Symbolic Constants and Identifiers have been eliminated by carefully structuring the syntax so that Symbolic Constants and Identifiers are never allowed to occur at the same place in the syntax.

The syntax has also been constructed so that the Name Spaces, of different "classes" of Identifiers, are separate. Thus, for example, a Cell Name can never clash with a Library Name, which can never clash with a Port Name.

For more information about the syntax of the EDIF standard, refer to the latest EDIF Specification [1].

The Semantics of EDIF

In the EDIF syntax the keywords and their associated constructs are used to represent electronic design data, such as: objects, their characteristics, relationships between them, and placeholders for grouping such information. The semantic definitions of the electronic design data that can be represented by the EDIF syntax are borne by the EDIF Keyword Names and their definitions, as presented in the EDIF Specification. The semantics definitions in the Specification draw from common terminology as used within the electronics industry. If, through user feedback, the semantics is ambiguous, then the definitions are refined to an unambiguous state.

Each translator writer must determine a mapping between his or her system's data representations and the EDIF format. By following a good Translator Development Methodology, this has proven to be a fairly simple task [6, 7].

Three Levels of Power

EDIF descriptions can be specified in three different levels of complexity: 0, 1, and 2. Level 0 is the basic level. It requires that all values be constants and allows no parameters, with the exception that parameters can be passed to external cells that are defined at levels 1 or 2. In level 1 cells can be defined with parameters, variables are allowed, and numerous functions and operators are available. Level 2 is the highest level currently defined. It provides control flow constructs. Level 0 is a proper subset of level 1, which is a proper subset of level 2.

Temporarily Extending EDIF

Problems can occur when no mapping can be identified between a data representation in the EDIF Writer's system and the EDIF Standard because the proper data format has not been defined in the Specification. The TC anticipated such a problem and provided a mechanism for the Standard's syntax to be extended, temporarily, by EDIF translator writers. This extension is accomplished by using the "userData" keyword and its construct. The form of the "userData" construct is:

(**userData** *KeywordNameDef* { *form* })

The construct consist of a left parenthesis, followed by the keyword name "userData", followed by a proposed keyword name (assigned by the translator writer), followed by zero or more EDIF "form" constructs, and finally followed by a right parenthesis. Obviously, before an EDIF Reader can read this construct, its meaning and format must be made known by the translator writer who created it. Any use of userData should be made known to the TC via a Users' Group so that future releases of the Standard can eliminate its need.

Organizational Structure of the Format

The overall structure of the EDIF Standard is described in detail in the latest version of the EDIF Specification [1]. A summary of that description is given in the following paragraphs:

Design information is organized in the EDIF in the form of a hierarchy, abstract at its higher levels, and becoming more detailed at lower levels. At the highest level of this structure is a single logical file. A file must contain certain status information, and may contain several designs and several libraries. A library contains a technology definition and several cell definitions. A design uniquely identifies a particular cell within a library as the top level of the design hierarchy, thus providing a starting point for the extraction of data from the file.

An EDIF file as described so far might look as follows, where ellipsis (...) indicate that further data is possible:

```
( EDIF ...
      ( status ... ) ...
      ( design ... ) ...
      ( design ... ) ...
      ( library ...
            ( technology ... ) ...
            ( cell ... ) ...
            ( cell ... ) ...
      ) ...
      ( library ... ) ...
)
```

The library provides a mechanism for grouping cells which share some common characteristics, such as a process technology or information for logic simulation. These characteristics, defined once in a technology block within the library, are associated with each cell that is also defined within that library.

The design hierarchy is expressed by including instances of other cells within a cell definition. For each cell defined in a library, different representations (called "views" by the EDIF) may be specified. For each representation the details (called "contents" by the EDIF) of the cell may be defined. Each view may also specify an "interface", which is a set of conventions defining the communication between that cell and other cells that instantiate it. The interface can be seen as a contract between the cell and its users and serves as an abstraction of the cell.

A cell definition, as discussed above, might be specified in the EDIF as follows:

```
( cell ...
      ( view SCHEMATIC ...
            ( interface ... ) ...
            ( contents ... ) ...
      ) ...
      ( view BEHAVIOR ... ) ...
      ( view MASKLAYOUT ... ) ...
)
```

The EDIF Specification defines the valid "view types." Currently they are MASKLAYOUT, DOCUMENT, BEHAVIOR, SCHEMATIC, SYMBOLIC, NETLIST, and STRANGER. The exact meaning of each view and the constructs that are allowed in each is specified in the latest EDIF Specification [1].

More Format Details

The details of an electronic is specified within three main EDIF constructs: technology, interface, and contents.

The Technology Block

Defined within the technology section are: the number system (numberDefinition), the common information associated with geometric figures (figureGroupDefault), design rules (physicalDesignRule), simulation values (simulationInfo), etc.

The following is an example of a technology section in EDIF:

```
( technology CMOS3micron
      ( numberDefinition SI
            ( scale distance 1 ( E 1 -6 ))
            ( scale time 2 ( E 1 -3 ))
      )
      ( define output figureGroup Poly )
      ( define output figureGroup Diff )
      ( figureGroupDefault
            ( figureGroupSpecification Poly
                  ( width 3 )
                  ( pathType truncate extend )
            )
            ( figureGroupSpecification Diff
                  ( width 3 )
                  ( pathType truncate extend )
            ) ...
      ) ...
      ( simulationInfo basic
            ( simulationValue t f z u ) ...
      ) ...
      ( physicalDesignRule
            interFigureGroupSpacing
            Poly_Diff Poly Diff
            ( ruleCheck
                  ( ruleRange ( at least 3 ))
            )
      ) ...
)
```

The Interface Block

Defined within the interface section are: ports (the objects through which signals are transferred between the instance hierarchy), schematic symbols, mask layout protection frames, simulation stimulus and expected response, pin permutability, etc.

An example of an interface section for a schematic view follows:

```
(view schematic SK
( interface
        ( comment "two input nand gate" )
        ( define inout port (multiple VCC GND ))
        ( define in port (multiple A B ))
        ( define out port (multiple C ))
        ( permutable A B )
        ( body symbol
                ( portImplimentation A
                        ( figureGroup symbol
                        ( dot ( point 0 40 ))))
                ( portImplimentation B
                        ( figureGroup symbol
                        ( dot ( point 0 20 ))))
                ( portImplimentation C
                        ( figureGroup symbol
                        ( dot (point 70 30 ))))
                ( figureGroup symbol
                        (path (point 0 40) (point 10 40 ))
                        (path (point 0 20) (point 10 20 ))
                        (path (point 65 30) (point 70 30))
                        (circle
                                (point 60 30) (point 65 30))
                        (shape
                                (point 40 10) (point 10 10)
                                (point 10 50) (point 40 50)
                                (arc
                                        (point 40 50)
                                        (point 60 30)
                                        (point 40 10 ))) ...
                ) ...
        )
) ... )
```

This example describes a two input nand gate cell, and its schematic symbol. The syntax follows a proposal for a new "body" construct that is under consideration by the TC [2].

The Contents Block

The contents section of the EDIF is where all the detailed implementation information is represented. These details may be actual, low-level descriptions of the cell's components, or they may be instances of other cells. The actual geometry of IC mask layouts is specified here, as well as the netlist which specifies the connectivity of the interior of a cell. Behavioral models are specified in the contents section, as is the placement and routing information of printed circuit boards, gate arrays, schematics, VLSI masks, etc. Of course, this information must be placed in the contents section of their view type.

An example of the contents of a cell follows:

```
(view schematic SK ...
(contents
        (comment "schematic representation of a two
        input nand gate")
        (instance depl_xter SK depl_xter_1
                (transform (translate 80 135)))
        (instance enh_xter SK enh_xter_1
                (transform (translate 80 90)))
        (instance enh_xter SK enh_xter_2
                (transform (translate 80 50)))
        (instance pow_conn SK pow_conn_1
                (transform (translate 80 170)))
        (instance out_conn SK out_conn_1
                (transform (translate 110 110))) ...
        (net VCC
                (joined VCC
                        (qualify depl_xter_1 D)
                        (qualify pow_conn_1 P))
                (figureGroup schematic
                        (path
                                (point 80 150)
                                (point 80 170))))
        (net OUT
                (joined C
                        (qualify depl_xter_1 S)
                        (qualify enh_xter_1 D)
                        (qualify out_conn_1 P))
                (figureGroup schematic
                        (path
                                (point 80 100)
                                (point 80 110))
                        (path
                                (point 80 110)
                                (point 80 120))
                        (path
                                (point 80 110)
                                (point 110 110)))) ...
) ... )
```

The above example is a partially complete schematic of a two input nand gate. The new net construct is currently being considered by the TC [2].

Design Information Coverage

The current version of the EDIF Standard can represent the following types of design data:

* Hierarchical Design Data.

* Schematic Symbol Libraries.

* Physical Design Libraries for PCBs, Gate Arrays, Standard Cells, and Full Custom VLSI.

* Design Data in the form of Schematics and Netlists.

* Physical Cell Placement and Interconnect Routing for PCBs, Gate Arrays, Standard Cells, and Full Custom VLSI.

* VLSI Mask Layout.

* Simulator Stimulus and Response Data.

* Simulator Logic and Timing Models.

The above list represents a large part of the total type of electronic design data in use today. Those areas that are missing, such as Physical and Electrical Design Rules, Procedural Design Techniques, Test Programs, High Level Behavioral Design Descriptions, etc., are being addressed by Technical Subcommittees.

Translator Development Methodology

Several translators have been developed within the past year and a half. From these experiences some design and development techniques have proven to be successful [7].

1. Identify the information that is to be translated.
2. Develop a formal Translation Specification.
3. Define all objects and relationships to be translated.
4. Map these objects and relationships into EDIF terminology and syntax.
5. Develop a proper subset of the EDIF syntax, which covers the data to be translated.
6. Develop a hand generated example of the data in EDIF format.
7. Review the specification with others who are familiar with the data to be translated and/or familiar with the EDIF Standard.
8. Implement the translator using the Translator Specification as a guide.
9. Test the new translator by writing data for, or reading data from, other existing translators.

Summary

The Electronic Design Interchange Format has become a viable data standard through which companies and institutions can exchange electronic design data. Although the EDIF is continuously being enhanced, it is now solid enough in many areas to be useful. Many EDIF translators have been written, with many more currently under development [6,8]. Those who would like to know more about the EDIF should obtain the current specification [1]; those who would like to contribute to future enhancements of the EDIF should contact the EDIF Users' Group.

Acknowledgements

The EDIF Standard is the result of the work of many individuals and their sponsoring companies or institutions. Although the number is too many to mentions here, a few should be recognized.

The Steering Committee:

Gordon Adshead, ICL, (European Representative); John Crawford, Tektronix, Inc.; Mike Feuer, Mentor Graphics Corp.; William Johnson, Texas Instruments, Inc.; Jeff La Vell, Motorola, Inc.; Richard Newton, U.C. at Berkeley; Robert Rozeboom, Texas Instruments, Inc.; Dick Smith, National Semiconductor , (Chairperson); Tony Zingale, Daisy Systems Corp.

The Technical Committee:

John Crawford, Tektronix, Inc., (Chairperson); John Eurich, Independent Consultant, (past PDR TSC Chairperson and Steering Committee member); Susan France, Mentor Graphics Corp.; Sanjiv Kaul, Daisy Systems Corp.; Richard Newton, U.C. at Berkeley; Mike Ransom, National Semiconductor Corp.; Paul Stanford, Texas Instruments, Inc. (Users' Group Chairperson); Mike Waters, Motorola, Inc.

Technical Subcommittee Chairpersons:

Keith Bennett, STL, (UK User Group); Dave Burson, Texas Instruments, (Printed Circuit Board TSC); Bill Denbeste, Test Systems Strategies, (Test TSC); Hewig Heckl, German Univ., (Germany User Group); Allan Kuchinsky, Hewlet Packard, (Procedural TSC); Ellen Mickanin, Mentor Graphics, (Behavioral TSC); Ed Prestwood, Motorola, (Public Relations Cmtg.); Lou Scheffer, Valid Logic, (Physical Design Rule TSC)

Special recognition is due to Sally Hill of Motorola, Inc., for coordinating the many meetings of the EDIF committees.

References

[1] EDIF Steering Committee, Electronic Design Interchange Format Version 1 1 0, 1985.

[2] EDIF Technical Committee, EDIF Proposals, ICCAD EDIF Workshop, Nov. 18, 1985.

[3] EDIF Steering Committee, Electronic Design Interchange Format Version 0 0 8, 1984.

[4] EDIF Steering Committee, Electronic Design Interchange Format Version 1 0 0, 1985.

[5] J. D. Crawford, "An Electronic Design Interchange Format," Proc. 21st. ACM IEEE Design Automation Conference, June 1984, pp. 683-685.

[6] J. P. Eurich, "Experiences Using EDIF," Proceedings of ADEE-East, Oct. 1985, pp. 338-343.

[7] J. P. Eurich, "Hierarchical Netlists in EDIF," The First EDIF Users' Group Workshop Digest of Technical Papers, June 27, 1985, pp 2-12.

[8] M. Feurer, "Gate Array Design with EDIF: A Case History," Proceedings of ADEE-East, Oct. 1985, pp. 333-337.