The Electronic Design Interchange Format EDIF: Present and Future

Hilary J. Kahn
Dept. of Computer Science
University of Manchester
Oxford Road, Manchester, M13 9PL UK

Abstract
This paper reviews the present status and current developments of the Electronic Design Interchange Format, EDIF. After a review of the role EDIF has at present, some of the new ideas being developed for inclusion in the new release of EDIF (Version 2.1.0) are introduced.

1: Introduction
Work on the definition of the Electronic Design Interchange Format EDIF started in 1983 and the first balloted version of the EIA standard, EDIF Version 2.0.0, was published in 1987. It became an ANSI standard as well in 1988 [1]. At the time, EDIF offered a way forward for end users and developers alike. For the end user in particular, it gave hope that there really could be a choice of design tools for different design stages (say) by transferring design information between different CAE systems, and the route to silicon or manufacture in general would have choices because a neutral format for transfer was available.

The original enthusiasm for EDIF assumed that it would be feasible - both technically and commercially - for CAD vendors to implement conformant transfer software. The reality has turned out to be rather different in some respects. It is true that the EDIF format is not without flaws; however the format is considerably more powerful and applicable than much current CAD vendor software permits. There have unfortunately been failures to understand the format and somewhat liberal interpretations of its semantics.

Despite problems, EDIF is now the primary mechanism for transfer of connectivity and schematic data. However, real practical problems exist and are being tackled. In this paper some of the difficulties that have arisen are reviewed and some solutions are discussed. The paper also summarises some of the extensions that are to be available as the next release of EDIF.

2: EDIF Version 2.0.0: Technical Status
EDIF Version 2.0.0 is intended primarily as a mechanism for the transfer of design information between CAD systems or from CAD system through to manufacture. At the time when it was first developed, the main interest in the format was to be found among silicon houses and CAD vendors whose products supported design on silicon. It followed therefore that the capabilities of EDIF Version 2.0.0 should have been focussed on meeting the needs of transfers of interest to ASIC designers.

EDIF offers a very structured representation of design data. The information is gathered into a set of libraries, which will typically hold information which is related by use (e.g. a library of schematic symbols) or by technology. Within a library the information is structured into a set of cells which are the main unit of design. Within a cell, the information is further structured into a set of views. The allowable types of view relate closely to the stages of the design process and hence directly to the points in the design process where design transfer may be expected. In EDIF Version 2.0.0, the main design related views are:

- NETLIST. This captures connectivity information together with aspects of timing specification.
- SCHEMATIC. This allows schematic symbols to be defined and provides for the description of multi-page schematic diagrams. The underlying connectivity represented by the schematic may also be captured. Again, timing information may be included.
- MASKLAYOUT. This captures the geometric layout of an IC together with the underlying connectivity.
- SYMBOLIC LAYOUT. This is intended to capture layout information and connectivity in terms of an abstract symbolic representation instead of mask level layout
- LOGICMODEL. This type of view allows the behaviour of a cell to be described using an event driven modelling representation.
- PCBLAYOUT. This view does not have adequate capability in EDIF Version 2.0.0. It is intended to support the definition of printed circuit board geometry and the underlying connectivity it represents.

An important characteristic of the EDIF design representations is that they are all capable of capturing the details of design hierarchy. This means that an EDIF description can accurately capture the instance and definition hierarchies found in databases which support electronic CAD processes.
The industrial take-up of EDIF Version 2.0.0 has been widespread in the United States, Japan and Europe and active user communities exist throughout the world. The EDIF views which are most widely supported are Netlist and Schematic. These domains allow EDIF to be in regular use not only for the transfer of design data from one CAD system to another, but also to allow 'point tools' to be linked into larger CAD systems to give additional functionality. EDIF is in common use passing design information to or from tools such as simulators, partitioning and placement algorithms, analysis tools and netlist extractors. In addition to its intended role linking CAD systems, EDIF has also been used to provide the nucleus of the inter-tool communication links that are now being addressed by initiatives such as the CAD Frameworks Initiative (CFI).

Practical problems have however been identified in using EDIF. In many cases these are the result of inadequacies in the EDIF Version 2.0.0 and some of these issues are addressed in the rest of this paper. Other problems have arisen as a result of the inadequacy of the translators produced by some CAD vendors. The ultimate goal of EDIF should be to become invisible - to be the unseen link that moves design information across distributed systems or between totally disjoint systems. The reality at present is that there are EDIF translators which fail to conform to the standard or which are selective in the way in which they process EDIF information. It is still possible to find on the market, CAD systems which are even unable to read in the EDIF that they themselves generated!

3: EDIF Methodology

The approach adopted in the EDIF organisation to identifying problem areas in EDIF has involved:
- gathering comments and requirements from active EDIF participants in EDIF Technical Subcommittees
- providing a technical Query Answering Service. This service, in addition to providing answers to individual technical queries (and publishing the questions and official answers for other users [2]), has highlighted ambiguities, misunderstandings and extension needs in EDIF Version 2.0.0.
- gathering review comments and requirements from users and developers of EDIF interfaces through published papers and Users' Group workshops in the US, Europe and Japan.

In order to ensure that problems were correctly understood and to help define solutions unambiguously, the EDIF Technical Committee decided to adopt a formal approach to the specification of proposed extensions to EDIF. Early in 1990, it was agreed that an Information Model written in the information modelling language EXPRESS [3] would be a prerequisite for reviewing or accepting any extensions to EDIF. This decision, which followed a review by the EDIF Technical Committee of alternative formalisms, has meant that before new proposals for EDIF syntax are generated, an agreed information model must be accepted. Appendix 1 outlines briefly what information modelling is and how this is achieved in EXPRESS.

This decision is a formal recognition of the real nature of EDIF. First and foremost EDIF has an inherent understanding of the structure and semantics of electronic design. It is built on a model of the world of electronic design; until now, however, that information model had not been written down in a way which is easy for engineers and tool developers to use. Secondly EDIF, as a transfer format, must have a syntactic representation to allow systems to generate and process files. This syntax complements the underlying information model and happens to use a lisp-like syntax which is easy to parse and to extend.

The decision to adopt information modelling has been successful in a number of ways. Firstly, it has meant that extension proposals come to the EDIF Technical Committee in the form of requirements specifications rather than suggested syntax which, experience showed, often took the form of overloading existing keywords. Secondly, it has meant that proposers of extensions have gone through a careful rationale creation process as they have developed their information models.

Following the decision to adopt information modelling in EXPRESS, there have been various activities involving the modelling of aspects of EDIF. A partial EXPRESS model of the current standard, EDIF Version 2.0.0, has been published [4] and further modelling work is being done by a special EDIF Information Modelling Group. The EDIF Technical Committee has developed an information model for enhanced connectivity representation and schematics [5]. This model was used as the basis for the syntax creation for EDIF Version 2.0.100 [6]. A more complete model covering all aspects of Level 0 EDIF is to be published in mid-1992. This is the model which forms the basis for EDIF Version 2.1.0.

EDIF Technical Subcommittees have also been very active in using information modelling in EXPRESS as the basis for their new proposals. Comprehensive models for EDIF PCB [7] and a new EDIF Test View [8] have been completed. These have been used for the development of draft syntax and interchange trials are under way.

Two additional benefits in the general electronic design domain have arisen from the use of EXPRESS information modelling by the EDIF organisation. Firstly, the availability of advanced models provides major technical input to CFI developments; the CFI Design Representation work is now in a position to draw directly on the work done in EDIF. They can take the EDIF information models and use them as the basis for the syntax crafting of the CFI Programming Interface. This will help to ensure that end-users find a more unified approach to the handling of design information in CAD systems.
A second advantage of using EXPRESS is in the international standards arena. EXPRESS is the information modelling language developed within the ISO Standard for the Exchange of Product Data initiative (STEP). A requirement for a product description mechanism to be considered for inclusion within STEP is that it should be supported by a formal model in EXPRESS. EDIF is at present being considered by the ISO/IEC Working Group on Electronic and Electrical standards within STEP.

4: EDIF Appraisal

An evaluation of the successes and failures of EDIF has indicated a number of technical areas where the EDIF Version 2.0 standard is inadequate or ambiguous. Some of these areas are reviewed here, followed by an outline of the approach being adopted to solve the problems raised.

The goals of the changes which have been made to produce EDIF Version 2.1, the EDIF release in preparation, include:
- More accurate transfer of design information. This has focussed both on ensuring that the new version of EDIF follows even more closely the concepts needed to represent design and on making it easier for CAD tool developers to provide accurate EDIF translators.
- Correction of known problem areas and removal of ambiguities found in EDIF Version 2.0.
- Specific focus on improving the handling of Connectivity and Schematics transfers.
- Providing a sound basis for the future extensions which include the proposed PCB representation and test representation.

5: Connectivity

The ability to represent connectivity is a key issue in the handling of electronic design data. In EDIF Version 2.0, the ability to represent single-bit connectivity is good. The main connectivity object, the net, is used to capture a number of concepts:
- In a net, there is a formal definition of which ports are joined (i.e. made electronically common) by that net.
- In the case of views which also carry implementation information (e.g. schematic or maskLayout views), the implementation graphics can be closely bound to the net definition.
- Nested structuring of nets can be captured to allow information such as routing criticality or timing details to be defined for parts of a net or to further bind pieces of implementation geometry to particular sub-nets.

The problem with EDIF connectivity arises in relation to the representation of busses and bundles. These terms, which are themselves ill-defined, typically imply connectivity which is wide or multi-bit rather than single-bit. In EDIF Version 2.0, there are two different ways in which wide connectivity can be captured: the arrayed net or the netBundle. These two mechanisms are quite different in nature: the arrayed net is in all respects analogous to a single-bit net, except that the number of wires conceptually involved is greater than 1; the netBundle on the other hand is a wrapper for a set of net definitions. It does not itself establish connectivity nor does it indicate which ports are to become electrically common. The nets defined inside a netBundle (which may themselves be arrayed) establish the detailed connectivity.

Furthermore, although it is true that for simple netlists all the connectivity at a given level of hierarchy is defined in one place, this is not true for more complicated cases. For example, if electrically common information happens to be described on multiple pages in a Schematic view, then in EDIF Version 2.0 there is no way of allowing the EDIF writer to indicate the total connectivity in one place. In addition, where information from a wide 'bus' is 'ripped' into other less wide nets or 'busses', a special type of cell is required. This means that a CAD system accepting a design in EDIF Version 2.0 must unravel the artificially inserted ripper 'cells' to deduce the underlying connectivity.

5.1: Proposed Solution for EDIF 2.1

The problem of representing busses and bundles is resolved in the next release of EDIF by recognising that, in common usage, busses and bundles both represent wide connectivity, but typically one form (the bus) allows constituents to be referenced by index (analogous to membership of an array) whereas members of bundles are commonly referenced by name. In EDIF 2.1, both forms of wide connectivity will be coalesced in a bus, with support for reference to constituents both by position (index) and name.

The proposed solution to capturing all the connectivity at one level of the design hierarchy is to introduce into EDIF the concept of Signal. All views that carry connectivity information (e.g. netlist, schematic, pcbLayout, maskLayout and symbolicLayout views) are divided into two sections, the logical connectivity section and the structure section. In the logical connectivity section, the abstract view-wide connectivity is specified in terms of signals which are one bit wide and which join all the ports that are electrically common at the current level of hierarchy. In addition, in the logical connectivity section, signals can be defined. These may be multi-bit and establish a grouping of signals. Signals may themselves be structured in terms of nested SignalGroups. These groupings provide a basis for the structured connectivity described by busses.

Figure 1 illustrates a grouping of signal information. In EDIF Version 2.1 this grouping would be reflected by having a SignalGroup instruction which grouped
signalGroups operator, operand and address. Eventually, signalGroup address, for example, would itself be defined as a grouping of signals, ad_0 through ad_15. In the structural part of the view, it is likely that signalGroup address would itself be represented as an address bus. The members of the bus may be referenced either by position in the bus (e.g. the 0th element of the bus called address) or by reference to their signal names (e.g. ad_0).

Figure 1: Grouping of signal information

In the structural connectivity section, nets and busses are defined. These allow the ordering of ports to be described. Each net is associated with a single signal and with single-bit ports. Each bus is associated with a signalGroup and with a set of ports of appropriate width.

Figure 2: Connectivity relationships

Nets may be further structured into subnets. The structuring into subnets allows specific attributes such as criticality to be associated with particular parts of a connection and supports the net implementation details that follow in views which include graphical implementation details (e.g. the schematic and layout views).

Busses may be subdivided in two orthogonal ways. They may be divided into subBusses or into busSlices. A subBus by definition has the same width as its containing structure. It is similar in nature to a subnet in that it allows additional structural information to be given about a part of the bus. A busSlice is used to define part of a bus that has a width less than the width of the immediately enclosing structure. A busSlice allows wide structures to be ripped into other less wide structures. Once again, the specific structure of nested busses, subBusses and busSlices used is determined by the need to specify attributes on parts of the structure and the requirements of the graphical implementation that may follow. Figure 2 illustrates the relationships between logical signals and signal groups, structural nets and busses, and implementation information.

6: Characteristics of re-useable objects

In EDIF Version 2.0 there is only one way of defining the re-use of an object via the instantiation process. This is the instance which may be applied to views of a cell. It followed therefore that the need to allow the re-use of a 'dumb graphics' object led, in EDIF Version 2.0, to the definition of a GRAPHIC view type which in all respects other than re-useability bore no resemblance to a 'true' cell view.

The Schematic Technical Subcommittee in developing their requirements for the next release of EDIF identified a number of object types which have the characteristic that they are defined once and re-used many times. Examples of these are off-page connector symbols and on-page connector symbols. These needs could have been met by continued extension of the EDIF view, but this would have resulted in the gross overloading of the view concept. Instead, the concept of 'Reusable Object' has been recognised and the needs of the Schematic Technical Subcommittee met by defining a sub-class of reusable objects called 'Instantiatable Objects'.

The goals of the definition of reusable objects of various types are to:
- differentiate clearly between different classes of re-useable object. The three classes identified are: views, view-related instantiatable objects and view-independent dumb graphics.
- clarify the definition of the semantics of a cell view.
- allow re-useable dumb graphics objects to be defined and used.
- allow objects which have special significance in a given type of view to be defined and re-used. At this time, the examples considered all come from the schematic view, but the concept is being extended to views of other types.
6.1: Characteristics of Cell Views

The instantiation of a cell view implies that a design object has been included in the design at the point of instantiation. A cell view has, or may be given, ports and has the potential for elaboration at a lower level of hierarchy. It is often a functional unit. A cell view may contain within it occurrences of other re-useable objects. Depending on the context (e.g. whether or not the type of view supports graphical information) these re-useable object occurrences may be instances of other cell views or occurrences of instantiatable objects.

7: Structuring of EDIF Views

In common design practice there are various ways in which a unit of design (EDIF cell) may be represented. One natural way is to use a symbol in a schematic to represent the existence of an instance of a unit of design at some point in the design hierarchy. When thinking about the cell as a `black box', the interface is the representation considered. At the point at which the design details are explored more fully - say to do a simulation or to complete a layout, then the contents of the design description are elaborated. That information is represented in EDIF as the contents of a particular type of view.

In EDIF Version 2.0, these representations were very closely interlinked, so a view contains an interface definition and a contents for elaboration, and certain types of view additionally contain a symbol definition. This model is not close enough to the way in which many CAD systems operate using symbol libraries (and hence to the way in which their users approach the design process). As a result, in the new release of EDIF, the relationships between these representations have been rethought.

In the planned version, views are grouped together into clusters which share a common interface. Within a cluster, there may be different types of view and various symbols which can be used as abstractions of the views. Because it is not realistic to require all views of a cell to share one interface (e.g. a logical connectivity view would not include the power ports that are essential for a layout view), a cell definition may contain multiple clusters.

Furthermore, to help capture the meaning of having multiple views of a given cell, a new viewGroup mechanism has been introduced. This allows views which are related for some user-defined or design configuration reason to be specifically identified. An indication can be given to show that one view is a new version of another, or that one view (say a flattened netlist) is derived from another view (say a hierarchical schematic).

8: Translator Support

The acceptance of EDIF as a truly viable design interchange format has been hampered by the poor quality of much of the translator software available to end users. In part, this has been the result of the fact that EDIF Version 2.0 offered considerable flexibility and includes some complex rules. In order to help overcome these problems, the new release of EDIF ensures that where possible a single concept can be defined in only one way. Furthermore, the syntax has been extended so that it is more self explanatory and explicit. More keywords have been introduced into the format - but the vast majority of those keywords are so structured that their use is limited to single contexts. This means that an EDIF reading tool will have the minimum interpretation to do in order to
map the objects described in the EDIF file to the CAD system database supporting the design engineer.

9: Conclusions

A new release of EDIF has been built using the experience gained from the widespread use and abuse of EDIF Version 2.0. The new version has been designed to minimize problems of understanding for EDIF readers and now includes more comprehensive support for connectivity description and schematic diagram representation. This development has been based on a practical yet formal approach involving the use of information modelling. A new release of EDIF is available for public review. Its supporting documentation includes not only the necessary full Reference Manual and Syntax required for implementation but also a comprehensive Information Model which tool developers and others can use to understand the concepts now included in EDIF.

Acknowledgements

The authors would like to thank all the members of the EDIF Technical Committee for their unstinting efforts over the years. Hilary Kahn would particularly like to thank Alan Williams and Rachel Lau of the University of Manchester for their support and enthusiasm. The contribution made by Hilary Kahn is supported in part by the European Commission through the European CAD Integration Project (ECIP).

References

2. EDIF Version 2.0 0: Questions and Answers, Volumes 1 - 5, University of Manchester, UK
5. Electronic Industries Association An Information Model for EDIF Version 2.0 100, February 1992
7. EDIF PCB Technical Subcommittee Information Model of a PCB (EXPRESS) Version 9, January 1991
8. EDIF Test Technical Subcommittee Information Model of the EDIF Test Extension EDIF 2.0 32, October 1991

Appendix 1: Information Modelling

An information model is a formal way of defining the objects, relationships, attributes and constraints for a given 'Universe of Discourse' (or domain). When the ISO Modelling Language EXPRESS is used, the objects in the domain are defined as entities, and relationships are established between entities. EXPRESS includes powerful ways of defining the constraints that must be adhered to. They may be defined using simple statements or, in complicated cases, using rules and functions.

The EXPRESS language style is relatively easy to learn and to read; experience to date shows that tool developers and engineers learn it readily and rapidly become accustomed to reading information models. The main reason may be that the language looks very much like an ordinary programming language with which most potential model reviewers are familiar. The following example illustrates part of the formal model of an EDIF net showing inter alia that a net relates to one signal and has associated ports:

```plaintext
ENTITY generic_net
ABSTRACT SUPERTYPE OF (ONEOF (net, subnet));
  associated_signal : signal;
  connected_instance_ports : SET OF instance_port;
  connected_master_ports : SET OF master_port;
  criticality : integer_token;
......
END-ENTITY;
```

Appendix 2: EDIF Organisation

The Electronic Design Interchange Format is being developed under the auspices of the EDIF Subdivision of the Electronic Industries Association (EIA). A Steering Committee is responsible for general guidance and a Technical Committee is responsible for the overall technical content of EDIF. Proposals and domain related work are done in Technical Subcommittees which have active participation in the USA, Europe and Asia. Technical advice for potential users or developers of EDIF tools is currently available from an EDIF Technical Centre. General Information may also be obtained from the EIA at the address given below.

Contact Addresses

Electronic Industries Association, Attention Patti Rusher
2001, Pennsylvania Avenue NW
20th & I, Washington DC 20006, USA
Fax: +1-202-457-4985

or

EDIF Technical Centre
University of Manchester
Department of Computer Science, Room IT403
Oxford Road, Manchester, M13 9PL, UK
e-mail: edif_support@cs.man.ac.uk
Fax: +44-61-275-6289