A real world application of EDIF

by MICHAEL A. WATERS
Motorola SPS, Inc.
Mesa, Arizona

ABSTRACT

This paper describes the first application within Motorola of the Electronic Design Interchange Format (EDIF), which is used to exchange Macrocell Array design information. The software based on this development is in regular use to support Motorola’s Macrocell Array products in the marketplace. The paper examines some of the issues involved, including both the design library and the completed design portion of the problem, illustrating the solutions that EDIF provides. An actual example from one of Motorola’s Macrocell Array libraries and a small design example using those libraries are used to show how the relevant information is expressed using EDIF.
INTRODUCTION

This paper describes the first use within Motorola of the first release of the EDIF standard, version 1.0. The application is oriented to a mainframe based Macrocell Array design system, which was in use when the project began. In each case, examples are derived from actual library or design data used in commercial designs although in some cases numeric information has been altered to protect proprietary information. Copies of the actual design databases and reference documents may be obtained by contacting Motorola’s ASIC customer engineering group.*

Figure 1 shows the overall scheme used; EDIF is used as a bidirectional interface between Motorola and an external customer. In planning this interface, six distinct classes of data were identified—three types of library and three types of data—to describe the completed design. For convenience, this paper is organized along these same lines, describing first the three types of library data and then the three types of completed design data.

In many cases, cell names and other EDIF identifiers have been chosen to aid human understanding of the EDIF information. This information would not be used by a computer system other than as a unique identifier of some EDIF object. That is to say, if the library identifier “MOTOROLA_MCA2” was to be changed to some other unique character string, such as “FOO_BAR”, throughout the file, then the semantics of that file are unchanged.

The examples used in this paper were drawn from Motorola Macrocell Array libraries for semicustom integrated circuit (IC) design, but the techniques presented apply equally well to any component library that EDIF can handle. This can include all types of electronic components, semicustom ICs, full-custom ICs, standard part ICs, passive components, board level parts, or complete modules.

MACROCELL ARRAY COMPONENT LIBRARIES

A component library typically reflects the product offering of a particular manufacturer. It must include enough information for a designer to successfully use the part in a design and to order the correct part from the manufacturer.

For semicustom integrated circuits (ICs) such as Macrocell Arrays, this library becomes an integral part of the design and ordering process for the entire integrated circuit. Since the entire design must then be built by a single manufacturer, the library becomes the basis for the product specification as well. As a result, the accuracy and completeness of the library is a concern of both the manufacturer and the ultimate user of the CAE system. One of the earliest motivations for developing EDIF was to simplify and expedite this transfer of semicustom IC libraries from the manufacturer to the ultimate user.

One of the useful features of EDIF is the tree-structured data organization, which allows an EDIF user to quickly isolate the small portion of an EDIF file of interest for the task at hand, consisting of a LIBRARY, CELL, and VIEW organizational hierarchy. The power of this structure is that it reflects the natural divisions of design information by source and intended use.

Note that the term VIEW implies there is some relationship between different VIEWS of the same cell (although EDIF does not force this), and that the actual object being described by each VIEW is the same object with different information being expressed in different VIEWS. Any number of VIEWS of a cell may be defined; the user selects only those VIEWS of interest for the particular application being addressed.

*Motorola Semiconductor Products Inc., Application Specific Integrated Circuit Division, Customer Engineering Support, Box 20912, Phoenix, Arizona 85036
Motorola divides these libraries by EDIF VIEW as follows:

**SCHEMATIC view**
Schematic symbols of each macrocell.

**NETLIST view**
Macrocell internal interconnection of simulator primitives.

**BEHAVIOR view**
Basic models of simulator primitives.

**SYMBOLIC view**
Physical information needed to place and route a design. This view includes the appropriate base array information and data for each macrocell.

Component Library Revision Control

One of the concerns of a manufacturer that is maintaining libraries that are not under his direct control is to ensure that any design referencing these libraries uses current and valid information. If there is a problem, it is important to identify it as quickly and easily as possible so it can be corrected with minimum impact on schedules and budgets.

As a result, one of the first steps in checking a design sent for manufacture must be to check the software revision, EDIF revision, and data revision. EDIF supplies EDIFVERSION and ACCOUNTING fields within the STATUS for this purpose. Motorola uses EDIFVERSION for problem tracing and ACCOUNTING as a preliminary validity check of a design on receipt to be sure that every cell used in the design is valid. This is accomplished simply by returning these status fields as part of the EDIF file, which specifies the completed design, and then comparing a revision code within them to a master list maintained by Motorola.

Schematic Symbol Library

The terms related to schematic drawings are often used loosely within industry. To understand a schematic symbol library, it is important to understand the difference between a "schematic symbol," a "schematic drawing," and a "netlist" for our purposes:

- **netlist:** connectivity data ONLY (which may be extracted from a "schematic drawing")
- **schematic drawing:** both graphics and connectivity data; references a number of "schematic symbols"
- **schematic symbol:** a cell with graphics and "connect points" which may, or may not contain, a "schematic drawing" to describe its contents

The schematic level component library consists of schematic symbols for each macrocell. These symbols can be interconnected to form a schematic drawing for the complete design. Each port of the macrocell is defined as a connect point, both as a symbol, and by using a flag for the user's CAE system. Standardized graphics are defined to display the same symbol both on a workstation and on a drawing. Several property values are supplied for each port to define such items as loading factors and port names. Similarly, some macrocell related properties, such as macro name and function, are supplied for each macrocell.

Figure 2 shows the schematic symbol of a typical macrocell from Motorola's bipolar macrocell library as it would appear on a workstation. Figure 3 shows a sample of the corresponding EDIF file used to transfer this cell to the workstation. This particular macrocell is a 3-2-2 OR/AND function taken from the design manual for the M2500ECL Macrocell Array.

The information in the EDIF file is an electronic form of the data found in the design manual, but it also includes such added information as explicit designation of pin functions.

The basic structure of the file fragment starts at the CELL level, defining the CELL name, its STATUS, and the VIEW of interest. An INTERFACE section defines the external symbol to be used for the macrocell, while the internal schematic may also be defined in the CONTENTS giving as much
or as little data as required about the internal working of the macrocell. For this library, no CONTENTS were required: the INTERFACE provided all of the data required to use the cell in a semi-custom IC design.

Within the INTERFACE, each port of the cell is specified with the DEFINE and PORTIMPLEMENTATION constructs, which give direction, pin symbol, swapability, and display information for the port. The graphical information is an important part of the symbol and is found in one or more FIGUREGROUPS, a construct EDIF uses to identify figures with common attributes, such as layer or color. Finally, a DOT shape is used to define the actual location and shape of the port. EDIF uses similar mechanisms to specify graphical data, whenever needed.

The BODY is used to supply information about the actual graphics of the symbol as seen on a workstation, once again using FIGUREGROUPS containing a number of EDIF SHAPE constructs. This time, a FILLPATTERN specifies that the SHAPE is left "white," or unfilled. A BORDER is defined for the symbol, with a dashed line (BORDER-PATTERN).

Simulation Library

The simulation library is divided into two parts, corresponding to usage within simulators. First, a library of cells containing simulator primitive data (BEHAVIOR view) provides a model of each primitive modeled by Motorola's internal simulator. Then, the main macrocell library defines the macrocell functional connections (NETLIST view), which connect the simulator primitives to make a particular macrocell.

Since EDIF V100 does not support an adequate BEHAVIOR view for this purpose, the library contains only the NETLIST view, which interconnects primitives that must be defined outside the EDIF library. This is far from ideal but has proven workable as an interim step and has allowed gate level simulation libraries to be ported with minimal effort.

Simulation example

Figure 4 shows NETLIST view of the M212 library cell shown in Figures 2 and 3. The logic primitives may be found in the LOGCAP user guide; they essentially are Boolean gates with rise and fall delays for each input and output.

The CELL, VIEW, and STATUS are given as before, but the INTERFACE is used merely to DEFINE the ports seen by the user. The CONTENTS is used to describe the inner workings of the cell. First, a LOGCAP primitive is referenced by INSTANCE, with PARAMETERS specifying rise and fall time. Next, the ports of this primitive cell are connected by JOINED statements, either to external cell ports or to other ports within the cell or to instances of primitives. The QUALIFY is used to specify that the named port is the port in a particular instance, rather than the port of the cell being defined. Thus, the complete internal connectivity of the cell is modeled using commonly known primitives. This description provides a highly accurate reference for the receiver, who may then decide to generate more efficient models using his own system's native primitives.

Physical Library

Structurally, the physical level library is the most complex and the most specific to a given technology. For a Macrocell Array, the library consists of three parts: the base array, the macrocells and the packages available for that array.

Defining the base array

The base array consists of the preassigned components (e.g., transistors, resistors) which are prefabricated, ready for the customizing metal layers. In a Macrocell Array, these metallization patterns come from two sources: pre-designed cells or macrocells, which perform a specific function when placed at the appropriate location; and interconnect wiring, which connects the macrocells to make the design. The places where a macrocell may be used are called sites and have complex symmetry rules associated with them.

The base array cell first defines the sites allowed for placement of the various classes of macrocells using a special section of EDIF called SOCKET definitions. This section defines such information as placement rules, symmetry of the sites, and the location of sites on the chip.

Since the underlying array is quite complex, a hierarchical definition has been used, as shown in Figure 5, with the basic sites defined as cells (e.g., INTERNAL_SITE) containing the basic SOCKET definitions. These are then instantiated into quads of four sites (e.g., INTERNAL_QUAD_SITE). These, in turn, are instantiated into the base array, with a STEP to specify the placement of each array of quad sites on the underlying array. This results in a pattern that can be flattened to show a simplistic representation of the placement sites, or maintained as a hierarchy for a sophisticated placement algo-

---

Figure 4—LOGCAP NETLIST view of the M212 library cell (ACLOGLIB)
The individual macrocell

The macrocell library includes the placement site requirements for each macrocell, including any special restrictions (such as a macrocell which may occupy more than one site), and the orientation of the macro required to correctly occupy a site. This data uses a section of EDIF that is complementary to the SOCKET called a PLUG, and the relationships between different PLUGs and SOCKETs define the often complex rules for correctly using a given macrocell in a given site. Also defined in the cell are any macrocell related routing barriers, physical port names, port locations, and relevant data about internal port to port connections, if any. Symbols may be defined for display of the macrocell on a workstation during the placement process. These symbols are distinct from those defined for the schematic drawing, and are used only for the PHYSICAL view of the design.

Figure 7 shows the SYMBOLIC view of the M212 library cell from Figure 2. This view gives the information required to successfully place this macrocell on the base array and to route wiring to the appropriate pins. Note that the internal routing of the cell (CONTENTS) is not given since this is not needed to use the macrocell in the array.

The INTERFACE defines a BORDER for display purposes similar to the one defined for the schematic symbol. As before, the physical appearance of each port is defined in a PORTIMPLEMENTATION. But in a PHYSICAL view, this specifies a routing target for a router, in this case a DOT. Some connectivity is specified, showing that port YD and SYD are equipotential and that a router may wish to use this as a feed-through connection. Finally, the ARRAYRELATEDINFO section defines the correct placement information using SOCKETSETs to specify the required sites and the orientation of the cell if it uses the site.
THE COMPLETED DESIGN

The requirements and capability of Macrocell Array users varies widely—from a small customer that is a first time user of Macrocell Arrays to a large sophisticated customer that designs dozens of arrays a year. This disparity is reflected in the different levels of required interface to the manufacturer. The small customer needs to minimize the risk involved and will use its own equipment just to generate a schematic drawing of the part, extracting just a netlist from which the rest of the design work is done. The large customer usually has installed equipment and procedures to do most, if not all, of the design in-house, sending a netlist, test patterns, and even physical or macrocell placement and interconnect routing information.

The sections of this paper are organized to correspond to each of the types of information to be transferred by means of the EDIF description. These three information groups are:

Netlist: Connectivity information normally derived from the schematic drawing, including the macrocells used for the design and their interconnection.

Test Pattern: Simulation input and output, in terms of signals applied to, and expected from, each pin of the device.

Physical: Defines macrocell placement in the array, routing of wiring and other manufacturing related information.

Three EDIF LIBRARY level sections are required regardless of the information sent: DESIGN, EXTERNAL, and TECHNOLOGY.

The DESIGN construct specifies the “root” or topmost cell in the design hierarchy, and is used as a starting point when extracting information about a design. The VIEW NETLIST within this cell contains the netlist data, the VIEW BEHAVIOR contains the test pattern data, and the VIEW MASKLAYOUT contains the macrocell placement and interconnect routing data. In each case, VIEWNAMEs for these three VIEWTYPEs have been chosen to help a human reader to understand their usage rather than to give some semantic understanding to a CAD system.

The macrocell library used for the design is specified using the EXTERNAL construct. For example, the EDIF statement (EXTERNAL M2500ECL) specifies that the Motorola M2500ECL Macrocell Array library was used. This allows reference to the library and its contents without requiring that the complete library be sent with each design.

A TECHNOLOGY section must be present to give at least the technology identifier for a simple netlist, but may also be needed to add information such as the scale factor to be used for timing information. For example, (TECHNOLOGY MOTOROLA_MCA2) specifies that Motorola’s MCA2 technology is being used and that the technology specific scaling and definitions will be found in that library. In this example, the definition case refers to the TECHNOLOGY section in the M2500ECL library previously supplied by Motorola.

Netlist Information

A 4-bit binary counter is used as a design example to illustrate the way data is expressed in EDIF. As in the library examples, only the basic information required is shown for clarity. The example is called “MOTO2500,” the schematic drawing is shown in Figure 8, and a partial EDIF file of the netlist is shown in Figure 9.

Test Pattern Information

A “Test Pattern” refers to a sequential set of logic states applied to or expected from each signal named in the EDIF file during the test sequence or simulation run. The data supplied consists of a series of logic states High (H, HIGH, T, or TRUE), Low (L, LOW, F, or FALSE), Ignore (X), or high impedance (Z).

Referring to the MOTO2500 example in Figure 8, a functional test is performed using the waveforms shown in Figure 10. If inputs are driven to the states shown, then the outputs shown are to be expected. To transmit this information a new
Physical Structure Information

The physical implementation of this array requires that each macrocell be placed in a legal site in the correct orientation and that interconnect wiring be routed to connect the ports of the macrocells. The rules for this process were supplied in the physical library description shown in Figure 6. Referring again to the MOTO2500 example in Figure 8, yet another view is added to the EDIF file (as shown in Figure 12), and gives a complete specification of the three types of information required to correctly build this design.

CONCLUSIONS

Both the software described and EDIF V100 have proven useful for exchange of actual designs. Because of the limitations of both the first version of this software and of EDIF V100, the software described will be replaced by EDIF V200 based production software in the near future. Since these two versions of EDIF are not upward compatible, a potential user of the EDIF interfaces described here should check with Motorola to be sure of the current status of
Motorola's software before writing their interface software, and with the EDIF User Group for the latest information on EDIF.*

*EDIF User Group, 2222 South Dobson Road, Building 5, Mesa, Arizona 85202

REFERENCES

1. EDIF Steering Committee. EDIF Specification, EDIF – Electronic Design Interchange Format Version 1 0 0, EDIF User Group, 1985