1

Introduction

CADIC is a hierarchical top–down design system for VLSI circuits. It has been developed at the Lehrstuhl of Prof. Dr. Günter Hotz at the Department of Computer Science, University of Saarland, Saarbrücken, Germany. The development of CADIC is supported by the Deutsche Forschungsgemeinschaft (DFG) within the Sonderforschungsbereich 124 *"VLSI–Entwurfsmethoden und Parallelität"* since 1983.

1.1 Features of the CADIC System

In comparison to other VLSI design systems, the CADIC system possesses the following distinctive features :

Parameterization

The design level of CADIC especially supports the specification of whole families of circuits which can depend on a set of parameters. For example these parameters can be used to have operands of flexible bit length (the user can design an n bit adder) or generic descriptions of circuits (a sorting network for arbitrary types of elements).

The designer can use sub–circuits which depend on arithmetical expressions over the set of circuit parameters. As well as the sub–circuits, the user may parameterize the width of a bundle of wires (equivalent to busses). With the help of parameterization multiple descriptions of the same circuit with

different sizes can be avoided. The use of parameterized sub–circuits and wires makes it possible to define a class of even very large circuits by only a few schematic inputs.

Recursive Specification

Recursion is one of the most impressive properties of CADIC . This feature admits recursive design operations in a topological framework and makes it possible for a hardware designer to easily specify regular circuits in a very compact and easy way. Like the recursive definition of functions in programming languages, references on the calling sub–circuit itself are allowed. The base of a recursive description are sub–circuits with fixed values for the parameters.

As well as the sub–circuits the designer has to parameterize the width of bundles of wires within a recursive specification. For the situation that it is too difficult to derive an arithmetical expression for a bundle of wires which holds for all stages of the recursion, CADIC offers the designer a very easy and comprehensive way of parameterizing wires. It allows the designer to use formal variables, denoted as wire types, to describe the width of a bus. It is possible to use an expression containing both indicated variables and arithmetical expressions over the set of circuit parameters. Within the recursive specification the indices of these wire types can be arithmetical expressions over the set of circuit parameters.

It is very significant that the recursive description together with the use of wire variables enables a user to setup generic specifications of algorithmic structures as for example sorting networks or parallel prefix computation parts ([Bur94]). These circuits for example can be specified independent from the realization of the basic operation. Later they can be configured according to special needs. The basic operation of a parallel prefix computation circuit can be chosen in order to construct a carry look ahead adder or a leading zero counter etc. without changing any other input of the parallel prefix network.

With the help of this description method the designer can setup his own library of reusable circuit structures. By this CADIC makes the graphical design methods become as strong and flexible as the text design methods.

Hierarchical Representation

An important problem during the design of large circuits is to control the amount of generated data. In CADIC we take advantage from the fact that large circuits often contain very regular structures. This can be used to setup a very compact representation in the form of an internal hierarchical data structure which is given by a directed acyclic graph (DAG). The DAG is a folded tree structure which is constructed in a top–down process over all sub–circuits of the design. Its root node is the given circuit C. All sub–circuits of C are internal nodes of the tree, in which same sub–circuits appear only once. The leaves of the tree are the basic cells or sub–circuits which only contain wiring structures. To construct this data structure for the required circuit, its parameters, if it depends on those, must be set to fixed values. This means that from the whole family of circuits given by a parameterized description we will now select one special representative (construct a 32 bit adder out of the n bit adder family).

After the hierarchical data structure is built, the system calculates all arithmetic expressions in the parameterized description, including the indices of the formal variables. If the designer has used wire variables to denote bundles of wires, the system will now derive a set of equations from the hierarchical representation in order to compute a legal solution for the variables. If this last step is successful, the created hierarchical data structure serves as the basic structure for the integrated design tools.

Integrated Design Environment

In order to make the integration of newly developed design tools possible, the graphical design environment of CADIC is implemented like a workbench. The core of the system is a graphical user interface which is basic for all design steps. For special needs of certain design tools it can be

enlarged by tool specific graphical components.

The modular implementation of CADIC supports the configurability of the whole system. Each tool is given by its own object code libraries, which can be linked into the graphical surface. Prerequisite is that each tool creates a relation between its internal data structures and the central hierarchical data structure of CADIC or that it directly works on it. Beside this a design tool may also be a stand alone program. In this case it can be connected to the graphical surface via interprocess communication.

Based on this open hierarchical data structure, CADIC has formed an integrated design environment which manages the different steps of the design process. The system CADIC 3 0 includes integrated tools for

- logic simulation
- timing analysis
- layer assignment
- power supply
- place and route
- net–list converters

The hierarchical data structure of CADIC provides a basis, on which new tools can be developed and easily integrated.

Visualization and Navigation

The integrated design tools of CADIC can directly display the calculated results and design data within the graphical user input, such that a relation between the design data and the specification is created.

The visualization of the data in combination with the navigation through the circuit hierarchy enables the user to find critical parts of a specification, where the design tools compute bad results. For example you can easily locate such regions of a design, where a logic simulator indicates wrong results in the specification.

In order to enable the visualization of data, the types of design data are specified by a simple description language, from which a parser generates the necessary library of graphical functions, that is linked into the surface of CADIC . If a tool is directly integrated in the graphical environment it can visualize its design data by calling the functions from this generated library. If a tool is not integrated into the surface and works independently, the data can be transferred to the surface of CADIC by interprocess communication (IPC).

Compared with pure text–form representation, there are especially possibilities to graphically display data using timing diagrams, point diagrams, line diagrams or histograms, etc.

Conversion Interfaces

There exist a lot of conversion interfaces in CADIC to export a circuit specification in the net–list formats of commercial systems. Especially this means that the designer cat setup parameterized circuit specifications and extract special representatives in a standard exchange format. CADIC 3 0 can generate the following different net–list formats:

- Electronic Design Interchange Format (EDIF 2 0 0)
- Desgin Exchange Format (DEF) for TANGATE Placement and Routing System from CADENCE
- Genrad Hardware Description Language (GHDL) for HILO Simulation Tool from GENRAD
- StrukturBeschreibungsSprache (SBS) for Venus Design System from SIEMENS
- Xilinx Netlist Format (XNF) for XACT FPGA Development System of Xilinx

Thereby, the CADIC system could be used as graphical front end for many commercial VLSI design systems. By selecting a certain library of basic

cells the hardware designer can choose any required technology to realize his design.

1.2 The Structure of CADIC

The integration of the given features into a common graphical environment is shown in figure 1.1.

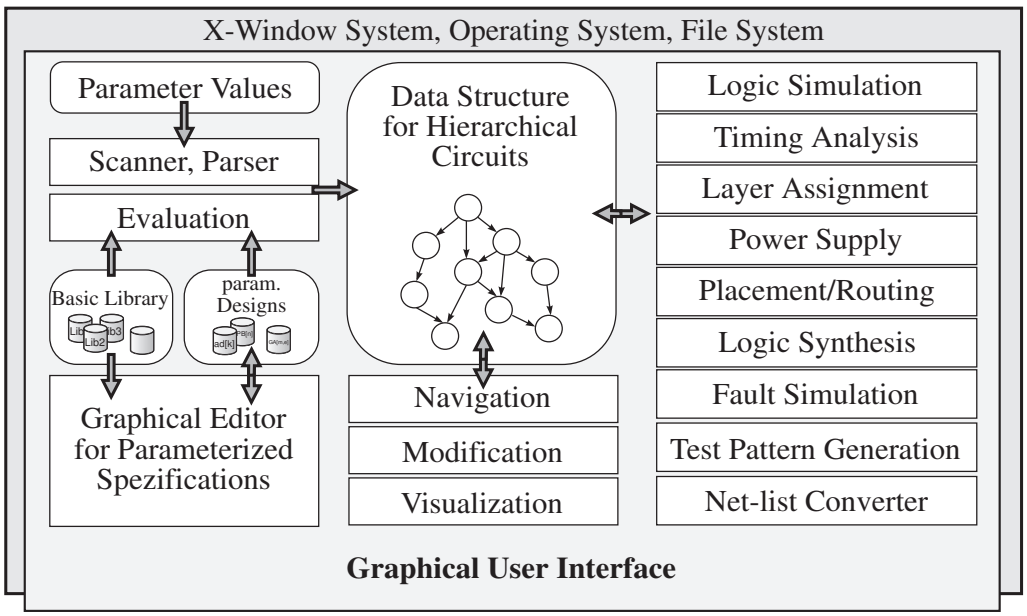

Figure 1.1: Integrated Design Environment of CADIC

With the help of the graphical editor the user can setup parameterized circuit designs over an arbitrary library of basic cells. From such a parameterized description the designer can select a specific element, i.e. a fixed circuit, by giving explicit values for its parameters. According to these parameter values the system will build a hierarchical data structure which is the base for all integrated design tools. This data structure is used to visualize the desgin results. With the help of navigation functions the hierarchical strucutre can be traversed in every direction in order to completely examine the design data.

1.3 The Structure of this Book

The structure of this book is orientated at the structure of the CADIC system as it is shown in figure 1.1.

In chapter 2 we will describe how the system can be installed on your hard disk.

In chapter 3 the main components of the graphical user interface and the setup of CADIC 's environment are explained.

In chapter 4 we will show, how you can setup a parameterized specification with the help of the graphical editor of CADIC . An adder for n bit operands is specified in order to introduce you to the work with the graphical editor.

In chapter 5 CADIC 's hierarchical data structure is explained. It is shown, how the system extracts one element of a parameterized circuit specification and builds up its basic hierarchical representation in the machine's memory.

In chapter 6 the logic simulation tool is introduced. This tools provides a simple method for checking your design on the base of the graphical specification, i.e. you can control the results of a simulation step graphically at the inputs and outputs of each subcircuit.

In chapters 7 to 9 we describe the integrated tools for layer assignment, for generation of the power supply nets and for calculation of a geometrical layout.

Chapter 10 gives you an impression of the parameterized design mehtod of CADIC . With the help of three examples we will demonstrate the power of this description method by giving short and handy specifications for even large circuits.

In chapter 11 CADIC 's interfaces to commercial systems and standard exchange formats are shown. With the help of these interfaces you can exchange a CADIC design with other design systems for further processing.

Chapter 12 contains a reference manual for the graphical editor. The basic syntactical structures for parameterized designs are explained as well as

the handling of the different editor functions. This chapter in addition to chapter 4 and chapter 10 enables you to setup parameterized descriptions yourself.

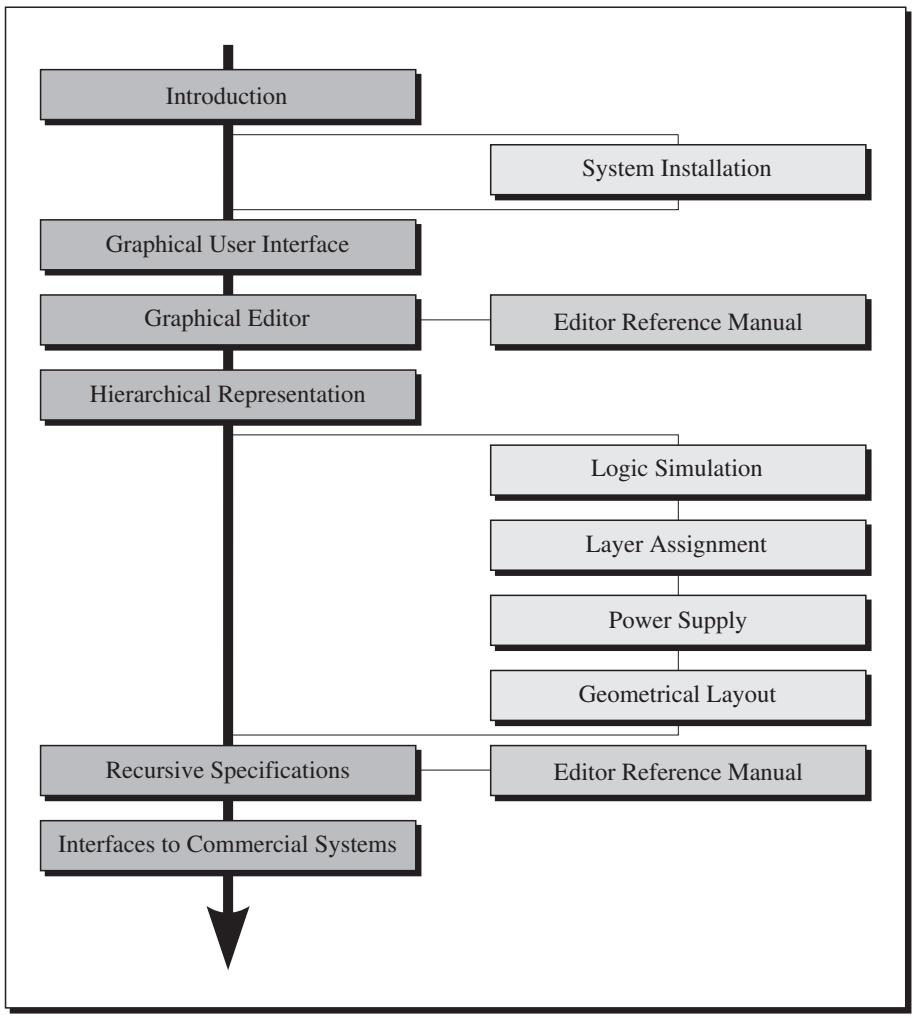

Figure 1.2: Readers guide for the CADIC manual

Figure 1.2 gives you a guide through the chapters of this book. For the understanding of the principles of CADIC you should read the main chapters (dark grey boxes) at least. The chapters about the system installation and the integrated design tools can be read later, when you want to perform the corresponding design task for a given specification. During the reading of the chapters about the graphical editor and the recursive specifications you can directly work with the system in order to setup these specifications

youself. In that case it may be helpful to have a look at the editor refrence manual.

1.4 Notations

In this book we use the following typographical conventions:

The *italic font* is used to introduce important new terminologies defined in the book which are used in following paragraphs.

The `typewriter font` is used to indicate system messages that appear on your screen. It also denotes user input to CADIC as well as system files, directories and environment variables.

Beside different fonts we use the following signs to give you some hints within the text:

This sign is used to mark important parts in the text. For example some requirements, without which the system would not work, are marked in this way.

⇝ 3,5.1

This sign gives you a reference to other paragraphs in the book, where you can find more information about the current subjects.

The extracted pictures from the menuline inserted in the text help you to identify the corresponding entries in the system menus and illustrate which button is currently to be selected. Within the text it is described how the corresponding function can be invoked and terminated.