This is to certify that the

thesis entitled

EXPLORING CADENCE EDA TOOLS FOR VLSI DESIGN

presented by

PETER L. SEMIG JR.

has been accepted towards fulfillment
of the requirements for

M.S. degree in ELEC. & COMPUTER ENGINEERING

[signature]
Major professor

Date August 3, 2001

EXPLORING CADENCE® EDA TOOLS FOR VLSI DESIGN

By

Peter L. Semig Jr.

A THESIS

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

MASTER OF SCIENCE

Electrical and Computer Engineering

2001

ABSTRACT

EXPLORING CADENCE® EDA TOOLS FOR VLSI DESIGN

By

Peter L. Semig Jr.

Massive growth in the microelectronics industry over the past decade has placed great importance on the VLSI design education of integrated circuit designers. There is a great need for new electrical and computer engineering graduates who have experience with industrial-standard VLSI design tools. Michigan State University currently uses a tool suite that is not well known throughout the industry. The purpose of this thesis is to demonstrate and document an approach for cell-based VLSI design using Cadence® EDA tools in conjunction with a standard cell library. The thesis can be transformed into a technical document with which students and faculty can gain experience with a recognized tool suite. This experience will make Michigan State University students more marketable and aid faculty research by providing a documented approach to manufacturing ASICs using industrial-standard EDA tools.

This paper is dedicated to my family.

ACKNOWLEDGEMENTS

First and foremost I would like to thank Dr. P. David Fisher for his unwavering support and strength. My family provided much needed support, strength, and understanding. I would also like to thank the following people for their contributions to the successful completion of this thesis:

- Mohammad Khalil, Malinda Funk, and Fred Hall (DECS). Administered and installed Cadence® tools and standard cell libraries. Provided tips when working with UNIX.
- Brian Wright & Roxanne Peacock (ECE Shop). Always willing to listen.
- Mark Johnson & Shawn Davidson (Purdue University). Demonstrated their design flow, provided documentation, and answered many Cadence® configuration questions.
- Wes Hansford & Glenn Jennings (MOSIS). Answered many questions about standard cell libraries. Also gave contact information of library vendors.
- Milos Backovic (LEDA Systems). Helped MSU obtain the LEDA standard cell library.

TABLE OF CONTENTS

APPENDICES

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AMI .... American Microsystems Inc.

AMIS .... AMI Semiconductor

ASIC .... Application-Specific Integrated Circuit

BCD .... Binary-Coded Decimal

CIW .... Command Interpreter Window

CMD .... Command File

CMOS .... Complementary Metal Oxide Semiconductor

CMU .... Carnegie Mellon University

CTLF .... Compiled Timing Library Format File

DAC2000 .... Design Automation Conference 2000

DECS .... Division of Engineering Computing Services

DEF .... Design Exchange Format

DFF .... Delay Flip-Flop

DK .... Design Kit

DSM SE .... Deep Submicron Silicon Ensemble

EDA .... Electronic Design Automation

EP .... Europractice

FF .... Flip-Flop

FPGA .... Field-Programmable Gate Array

FSM .... Finite State Machine

HDL .... Hardware Description Language

HP .... Hewlett Packard

IC .... Integrated Circuit

ICC .... IC Craftsman

I/O .... Input/Output

IP .... Intellectual Property

ITC99 .... International Testing Conference 1999

LDV .... Logical Design & Verification

LEF .... Library Exchange Format

LSW .... Layer Selection Window

MOSIS ...... Metal Oxide Semiconductor Implementation Service

MSU .... Michigan State University

NCSU .... North Carolina State University

NDA .... Non-Disclosure Agreement

NRE .... Non-recurring Expense

P&R .... Place and Route

PCB .... Printed Circuit Board

PSD .... PCB Systems Division

RTN .... Return

SCL .... Standard Cell Library

SIA .... Semiconductor Industry Association

SPW .... Signal Processing WorkSystem

TF .... Technology File

TLF .... Timing Library Format File

TSMC .... Taiwan Semiconductor Manufacturing Company

UMC .... United Microelectronics Corporation

VHDL .... VHSIC Hardware Description Language

VHSIC .... Very High Speed Integrated Circuit

VLSI .... Very Large-Scale Integration

INTRODUCTION

According to the Semiconductor Industry Association (SIA), the world has experienced a growth of approximately 400% in the sales of semiconductor devices during the past decade.[1,2] The vast majority of these devices (almost 90% and increasing) are integrated circuits (ICs or chips).[3] It is projected that the worldwide sales of semiconductors will reach over $300B by 2003, which represents over 600% growth since 1990.[3] This growth (and projected growth) has placed an emphasis on quick time-to-market, low power designs, low cost designs, and increased capabilities (or increased wafer densities). Powerful Electronic Design Automation (EDA) tools have been developed and are continually updated to try and meet these challenges.

The purpose of this thesis is to document and demonstrate the cell-based (or top-down) design flow associated with the Cadence® EDA tools. The thesis can then be transformed into a technical document, which can be used by other parties (students, faculty, etc.) as a reference for designing ICs with Cadence® tools. Chapters 1 and 2 provide introductions to the Cadence® cell-based design flow and standard cell libraries, respectively. Chapter 3 is a tutorial that demonstrates the design flow discussed in Chapter 1. The tutorial implements the 1999 International Test Conference (ITC99) b01 circuit. It is a finite state machine (FSM) that recognizes binary-coded decimal (BCD) numbers. There are 25 gates and 4 flip-flops (FFs) in the design. Chapter 4 discusses the plethora of lessons learned throughout the completion of this thesis. The final chapter, Chapter 5, concludes the paper by reviewing the accomplishments and providing suggestions for future work. The appendices include instructions for running the Cadence® tools and information about their associated files.

Additional reference material on VHSIC hardware description language (VHDL)[5,14], Verilog[6,9], UNIX[7,10], and complementary metal oxide semiconductor (CMOS) circuit design[8,11,12,13] are recommended preliminary reading.

CHAPTER 1

CADENCE® CELL-BASED DESIGN FLOW

1.1 *Introduction to Cadence®*

In 1988, ECAD, Inc. and SDA Systems merged to form Cadence Design Systems, Inc. (Cadence®). Today, Cadence® provides a suite of EDA software tools that are used to design leading edge ICs.

Cadence® makes available tools specifically designed for system-level design and verification, intellectual property (IP) reuse, functional/logical verification, custom IC design, digital IC design, IC package design, field programmable gate array (FPGA) design, and printed circuit board (PCB) design and verification. These tools provide the consumer with a complete IC design and test environment.

This thesis focuses on a subset of the tools provided by Cadence® to members of the Cadence® North American University Software Program. Michigan State University (MSU) has enrolled in the program and obtained all of the Standard University Program Bundles (Custom Integrated Circuits, Deep Submicron, Design and Verification, System Level Design, and PCB Systems).

1.2 *Design Methodologies*

Custom and semicustom are the two main digital circuit design methodologies. This thesis focuses on cell-based design, which is a type of semicustom design.

Custom design involves drawing the circuit topology and physical design at the lowest level. This method is used to maximize performance and/or design density. Drawbacks include a long time-to-market and high nonrecurring expenses (NRE).

Semicustom design, on the other hand, involves design at a higher level. Most of the circuit is usually specified in Verilog or VHDL and implemented using a standard cell library (SCL). Other areas of the IC are custom designed, and all parts are joined to make a semicustom IC. IC time-to-market and cost are greatly reduced with this approach. Even though this design methodology does not necessarily optimize performance or density, it is often the flow of choice.

1.3 *Cadence® Cell-Based Design*

Cell-based design using the Cadence® tools involves three main programs: Ambit® NaviGates®, Silicon Ensemble®, and Virtuoso®.

Ambit® NaviGates® performs circuit synthesis, Silicon Ensemble® places and routes standard cells, and Virtuoso® is the layout editor (Figure 1.1).

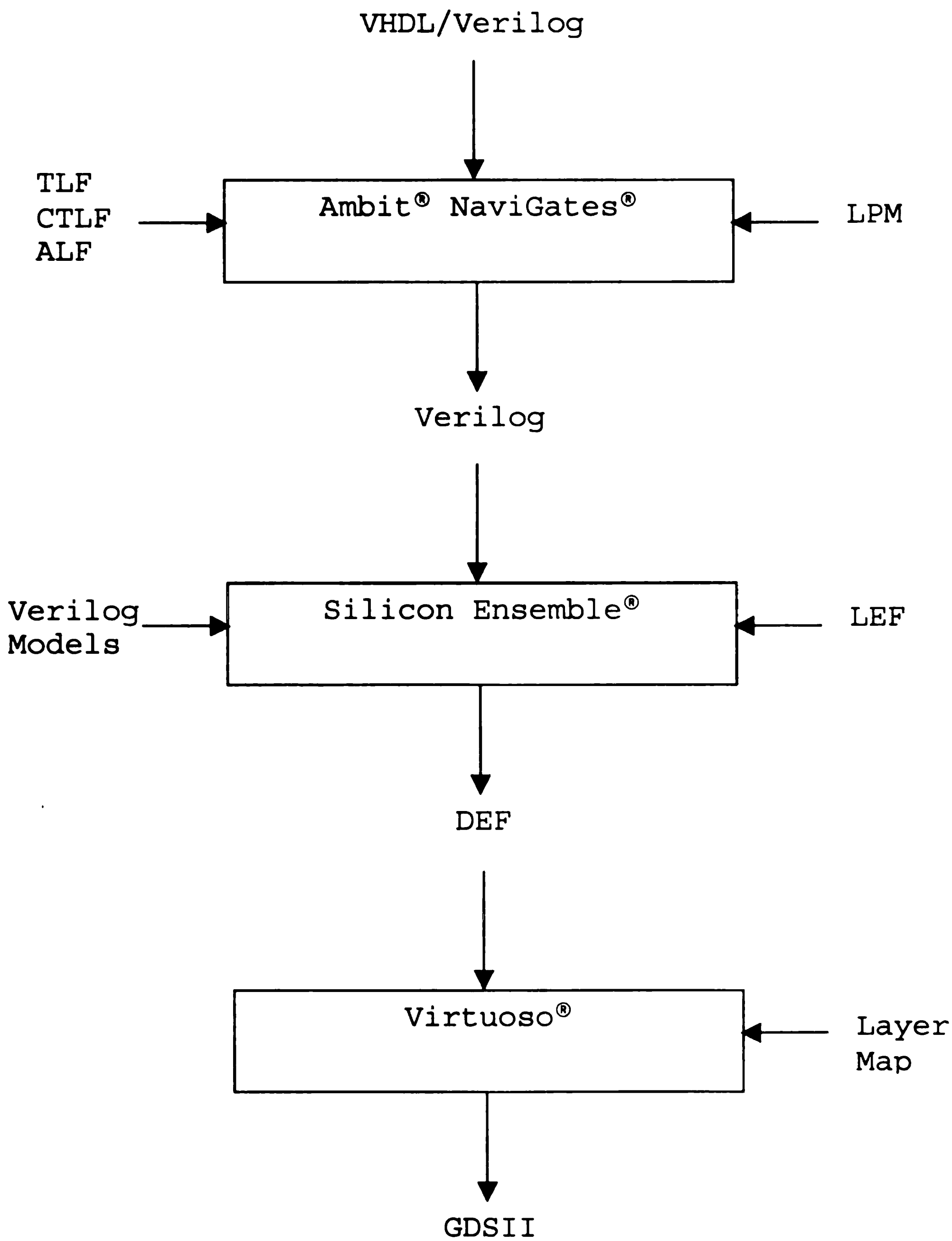

Figure 1.1: Cadence® Cell-Based Design Flow

Ambit® NaviGates® is located in the Ambit® tool set. Synthesis is the act of taking a design from an abstract description (or high-level) to the gate level. The high-level description is typically written in Verilog or VHDL. The circuit's inputs, outputs, and functionality are specified, but not the details of gate interconnections. Once the program reads in the hardware description language (HDL), a generic gate-level design is generated using standard parts. From here a specific technology can be attached to the design, and the gate-level schematic optimized for speed and/or area. For more information about the files associated with Ambit® NaviGates®, see Appendix B.

Silicon Ensemble® is located in the DSM SE 5.3 tool set. Silicon Ensemble® imports the Verilog description generated by Ambit® NaviGates®, SCL Verilog models, and SCL Library Exchange Format (LEF) file. Then the user generates the floorplan for the IC based on variables such as aspect ratio and row utilization. After the floorplan is complete, the input and output pins (I/Os) are placed. Placing the cells, refining I/O placement, power routing, standard cell placement, and final routing are the remaining steps. The IC can then be exported as a design exchange format (DEF) file for import into Virtuoso®. For more information about the files associated with Silicon Ensemble®, see Appendix C.

Virtuoso® is located in the IC 4.4.5 tool set, and is typically used closely with Composer® (the schematic entry tool), DIVA (the physical verification tool), and Analog Artist (the circuit simulation tool). For the purposes of this thesis we will only use Virtuoso®. Appendix D contains more information about the files associated with Virtuoso®.

After placing and routing the IC in Silicon Ensemble® the DEF file can be imported into a new library cell in Virtuoso®. The new library should be created using the appropriate SCL technology file in order to comply with all layer definitions, rules, etc. If supported by the library, custom layout can take place to join multiple designs. The final design can then be checked for design rule violations and exported as a GDSII file. The GDSII file can be sent to a silicon foundry for fabrication.

CHAPTER 2

STANDARD CELL LIBRARIES

2.1 *Overview*

In order to ensure a high quality very large-scale integration (VLSI) design education for students of MSU, it is of primary importance to demonstrate both the full-custom and semicustom design process. It is also crucial to have the student group's ICs fabricated. The key to achieving these educational goals is to obtain a SCL that supports both design methodologies in a technology that can be fabricated at a reasonable cost.

There are various small-volume IC production services that cater to educational institutions. Two of these services are the Metal Oxide Semiconductor Implementation Service (MOSIS) and Europractice (EP). While MSU has historically worked with MOSIS, they do not currently offer Cadence® compatible SCLs. MOSIS offers a few Cadence® compatible design kits (DKs) for their processes. DKs usually contain files for custom layout (technology, design rule checking, layer display, etc.) Some DKs contain a few basic standard cells, but rarely support standard cell place and route (P&R). EP, on the other hand, offers a plethora of SCLs for their processes that work with

Cadence®. In December 2000, Cadence® sent MSU's Division of Engineering Computing Services (DECS) the new release of their tools. However, the SCLs that were obtained from EP did not work with the new release.

There are also many companies (e.g. Artisan, Cadabra, and LEDA Systems) who design and sell SCLs for a variety of processes (e.g. UMC, TSMC). These SCLs are often expensive and occasionally not available to educational institutions to protect the company's IP. Even though MOSIS does not offer SCLs, they did provide us with a contact at LEDA who helped us obtain one of their SCLs. We were required, however, to sign a non-disclosure agreement (NDA) to protect their IP.

2.2 *Available Standard Cell Libraries & Design Kits*

A variety of SCLs and DKs have been obtained from MOSIS, EP, library vendors, and educational institutions for use with the Cadence® tools at MSU (Table 2.1). Since this thesis concentrates on cell-based design, SCLs will be the primary focus.

Each of the SCLs that have been acquired support a variety of tools and design methodologies. Artisan's United Microelectronics Corporation (UMC) 0.18 μm SCL, for example, supports standard cell P&R but not full-custom layout. The Carnegie Mellon University (CMU) library, on the other hand, has support for full-custom layout but lacks good P&R files. Finding one library that supports both semicustom and full-custom IC designs in a MOSIS or EP compatible technology is a major challenge. A library with all three requirements is essential to the success of MSU's VLSI design courses. Students who take a VLSI design course with MSU could then experience both design methodologies and have their chips fabricated for future testing.

Table 2.1: Cadence® Standard Cell Libraries & Design Kits

Acquired From	Foundry	Process	SCL/DK	Cost
MOSIS	Agilent/HP	0.5 µm	DK	Free
MOSIS	TSMC	0.18 µm	DK	Free
MOSIS	TSMC	0.25 µm	DK	Free
MOSIS	TSMC	0.35 µm	DK	Free
EP	Alcatel	2.0 µm	SCL	Free
EP	Alcatel	0.7 µm	SCL	Free
EP	Alcatel	0.5 µm	SCL	Free
EP	Alcatel	0.35 µm	SCL	Free
CMU	HP	0.35 µm	SCL	Free
Artisan	UMC	0.18 µm	SCL	Free
LEDA Systems	TSMC	0.25 µm	SCL	Free

Currently, MOSIS only supports the American Microsystems Inc. (AMI) ABN (1.5 µm) and AMI C5N (0.5 µm) technologies for their educational program. However, at this time MOSIS does not have Cadence® compatible SCLs available for these processes.

Removing the MOSIS DKs and EP SCLs from our options, three libraries are left. Their capabilities with regard to synthesis, P&R, layout, and fabrication were assessed (Table 2.2).

Table 2.2: Standard Cell Library Capabilities

	LEDA	Artisan	CMU
MOSIS Fabrication Process	No	No	No
EP Fabrication Process	No	No	No
Ambit Synthesis	No	Yes	Yes
Synopsys Synthesis	Yes	Yes	Yes
P&R	Yes	Yes	Yes
Layout Files	No	No	Yes
GDSII Export	Yes	Yes	No

It was found that the LEDA library lacked many of the necessary files to perform the complete design flow. While these files could be created, it is beyond the scope of this thesis. The Artisan library was found to have excellent P&R support, but layout files were lacking. The cell-based tutorial in Chapter 3 utilizes the Artisan library because of its strong P&R support. The standard cells for this library were not included with the library to protect Artisan's IP. The CMU library contained all the needed files, but there were problems associated with the P&R. Silicon Ensemble® placed cells in a manner that was not optimized. This inefficiency is probably related to the generation of the SCL's LEF file. Creating a new LEF file is also beyond the scope of this thesis.

2.3 *Standard Cell Library Issues*

It was found that the majority of SCLs have support for Synopsys's Design Compiler, not Ambit® NaviGates®. This provided a major problem when trying to take a design from HDL to gate layout. There is a utility, called "syn2tlf", which can convert a Synopsys LIB file to an Ambit® Timing Library Format (TLF) file. Unfortunately, there were issues with the version of the LIB file, for older LIB files were not compatible with the "syn2tlf" utility. Even if a TLF file was created from a LIB file, it was then found that our version of Ambit® does not support the import of a TLF file; it only supports the import of Compiled TLF (CTLF) files. This compilation of a TLF file proved yet another issue. The latest release of

Cadence® did not include the "tlfc" utility, which is used to compile TLF files. An old copy of the "tlfc" utility was obtained, but it did not compile the TLF files we have in our libraries due to version issues.

Another issue involved the import of the LEF file into Silicon Ensemble®. The original LEF file obtained with the CMU library was version 4.4. Silicon Ensemble® can only import version 5.1 or higher. An attempt was made to create a new LEF file of the latest version, but it was unsuccessful. A LEF file of version 5.1 was obtained through contacts made a Purdue University, but the placement of cells was not optimized.

CHAPTER 3

CADENCE® CELL-BASED DESIGN TUTORIAL

3.1 *Purpose*

The purpose of this chapter is to provide the reader with a tutorial that illustrates the Cadence® cell-based design flow.[4] The detailed objectives for each section are found in Table 3.1.

Table 3.1: Tutorial Section Objectives

Section		Objectives
3.2		Understand background information about the standard cell library used in the tutorial.
3.3		Understand the typing conventions used throughout the tutorial.
3.4	3.4.1	Learn how to synthesize a design using Ambit® NaviGates®.
	3.4.2	Learn how to P&R a design using Silicon Ensemble®.
	3.4.3	Learn how to import a design from Silicon Ensemble® into Virtuoso® and how to export a GDSII file.

3.2 *Background*

Artisan Components's UMC Gold Logic L180 Copper SAGE SCL was used for this tutorial. The feature size for this library is 0.18 μm and it utilizes copper interconnects. It is important to understand that the library does not contain the physical layouts of the standard cells. Also, it is lacking a display.drf file, which is used to identify layers in Virtuoso®. These and other parts are purposely removed from the library to protect Artisan's IP.

3.3 *Conventions*

The tutorial follows conventions consistent with Table 3.2.

Table 3.2: Tutorial Conventions and Examples

Function	**Description**	**Example**
Comment	Comments elaborate on the purpose of a particular step. They are identified with a '*'.	* This step routes power to all the cells.
Single Keystrokes	A single keystroke is made with the keyboard. The keystroke is characterized with **boldface**.	**RTN**
UNIX Command	A UNIX command is a statement entered via the keyboard at the '>' prompt of a terminal window. It will be distinguished with Times New Roman.	>source $SOFT/ic445 **RTN**
ac_shell Command	An ac_shell command is a statement entered via the keyboard at the 'ac_shell>' prompt in the bottom window of NaviGates®. It will be distinguished with Times New Roman.	ac_shell>source a.cmd **RTN**

Table 3.2 (cont'd).

Menu Selections	Menu selections are made by left clicking on the indicated word. These selections are in *italics*. Consecutive selections are separated by a '>'.	From the Toolbar, select *File* > *Save*
Specify Box	Some windows have multiple boxes that can be specified. **Boldface** type followed by a ':' will denote which box to specify. The information following is what to type or select in the given box.	**Selection:** /opte/cds/aci/ **RTN**
Tab	Some windows have tabs (Figure 3.1 has 8 tabs). When a tab is to be selected, ***bold italics*** will be used.	***Modules***
Double Click	A double click is two rapid left mouse clicks in a row. Underlining will distinguish a double click.	***Modules*** test *Double click on test in the Modules tab
Select	The term "select" means left-clicking once on the indicated item. The item can be either text or a button.	Select *OK*

3.4 *Procedure*

The procedure is divided into three main sections. Section 3.4.1 describes the steps associated with Ambit® NaviGates®; section 3.4.2 details the Silicon Ensemble®

procedure; section 3.4.3 explains how to generate the GDSII file using Virtuoso®.

3.4.1 *Ambit® NaviGates®*

The purpose of this section is to synthesize a design with Ambit® NaviGates®. We will copy all files needed for the tutorial, launch Ambit® NaviGates®, and execute a script file. The script file imports the Verilog design, synthesizes it with a generic technology, and optimizes the design with the UMC 0.18 µm technology. It then exports a new Verilog file that can be used by Silicon Ensemble® for P&R.

1) Logon to on one of DECS's UNIX stations.
2) Launch a terminal window.
    a. Right-click on the desktop.
    b. Select *Utilities* > *Terminal*.
3) Load the t shell. The t shell is a more sophisticated version of the c shell. Features include auto completion (using **TAB**), enhanced history, and command line editing.
    a. >tcsh **RTN**
4) Copy files for tutorial.
    a. >mkdir tutorial **RTN**

b. >cd tutorial **RTN** *Try >cd tu **TAB**

c. >cp /opte/cds/design_kits/tutorial/*.* ./ **RTN**

5) Launch Ambit® NaviGates® (Figure 3.1).

a. >source $SOFT/ambit **RTN**

b. >ac_shell –visual **RTN**

6) Execute command file from the ac_shell prompt in NaviGates®.

a. ac_shell> source syncommands.cmd **RTN**

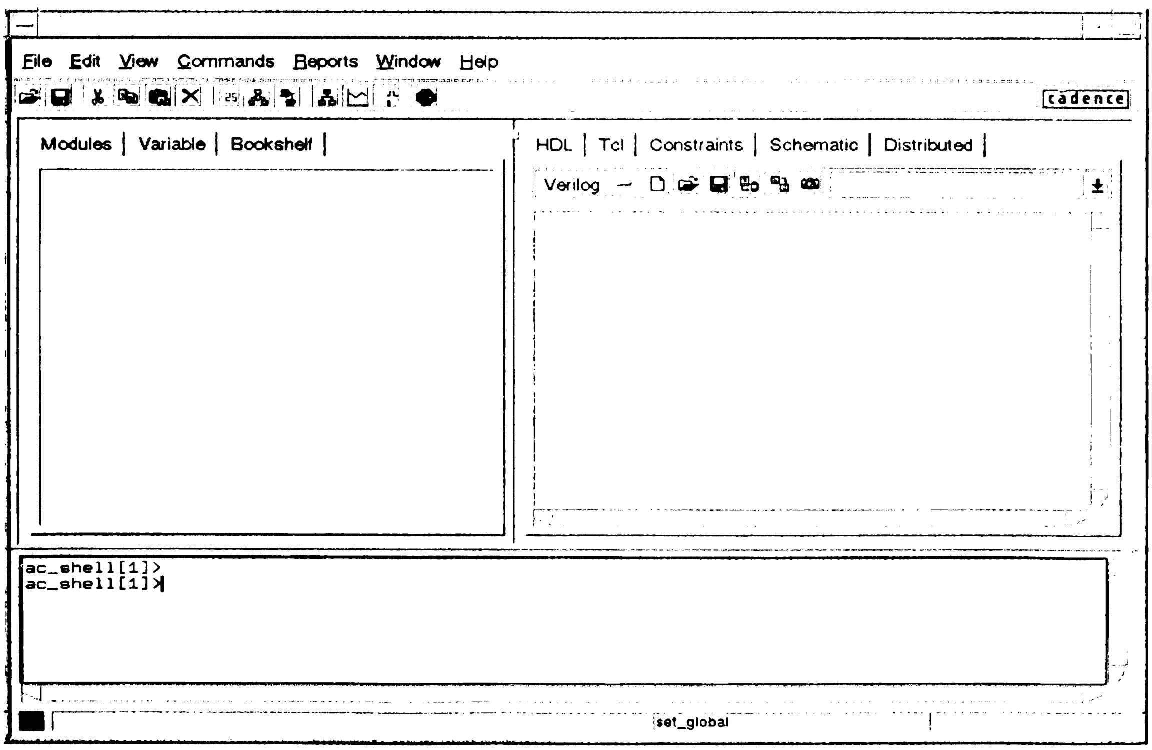

Figure 3.1: Ambit® NaviGates®

7) View schematic (Figure 3.2).

a. ***Modules*** b01 [b01] (m)

8) Exit Ambit® NaviGates®.

a. ac_shell> exit **RTN**

b. Select *OK*.

9) Close terminal window.

a. >exit **RTN** *Exit the t shell.

b. >exit **RTN** *Exit the terminal.

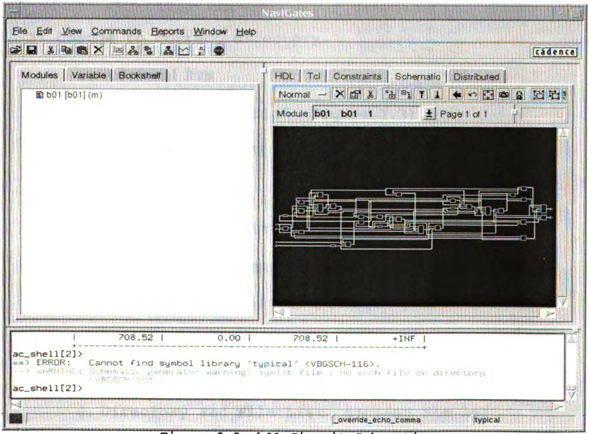

Figure 3.2: b01 Circuit Schematic

3.4.2 *Silicon Ensemble®*

This section will demonstrate how to launch Silicon Ensemble®, import the appropriate LEF file, import the Verilog description of the circuit and the Verilog models, floorplan, and P&R the I/Os and cells. A DEF file is then exported for use with Virtuoso®.

1) Launch a new terminal window.
    a. Right-click on the desktop.
    b. Select *Utilities > Terminal*.
2) Load the t shell.
    a. >tcsh **RTN**
3) Change to correct directory.
    a. >cd tutorial **RTN**
4) Launch Silicon Ensemble® (Figure 3.3).
    a. >source $SOFT/dsmse53 **RTN**
    b. >seultra **RTN**
5) Import LEF file (Figure 3.4).
    a. Select *File > Import > LEF*.
    b. **Selection:** /opte/cds/artlib/aci/sc/lef/ **RTN**
    c. **Options:** Select *Case Sensitive Names*.
    d. **Directory and File List:** Select umc18sc_5lm.lef.
    e. **Directory and File List:** umc18sc_5lm.lef

f. Select *OK*. *You should see "No errors found.

The database created successfully."

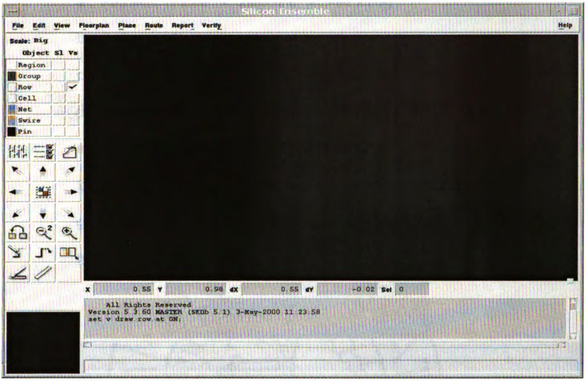

Figure 3.3: Silicon Ensemble®

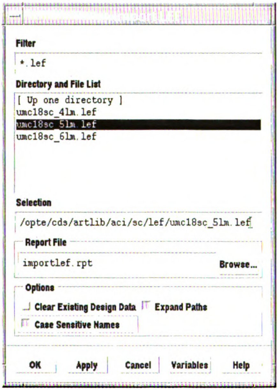

Figure 3.4: Import Library Exchange Format (LEF) File

6) Import Verilog (Figure 3.5).

a. Select *File* > *Import* > *Verilog*.

b. Select *Browse*.

c. **Directory and File List:** Select b01.v.

d. **Directory and File List:** b01.v

e. **Selection:** /opte/cds/artlib/aci/sc/Verilog **RTN**

f. **Directory and File List:** Select umc18sc.v.

g. **Directory and File List:** umc18sc.v

h. Select *OK*.

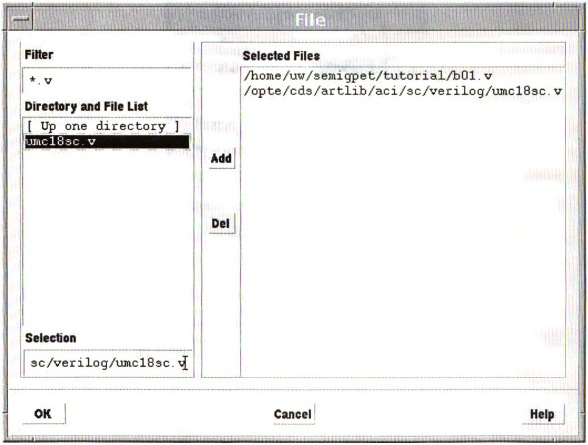

Figure 3.5: Verilog Source Files

7) Set Verilog Top Module (Figure 3.6).

a. **Verilog Top Module:** b01 *The 0 in b01 is a zero.

b. Select *OK*. *This step takes a few minutes. When finished you should see "End importing verilog."

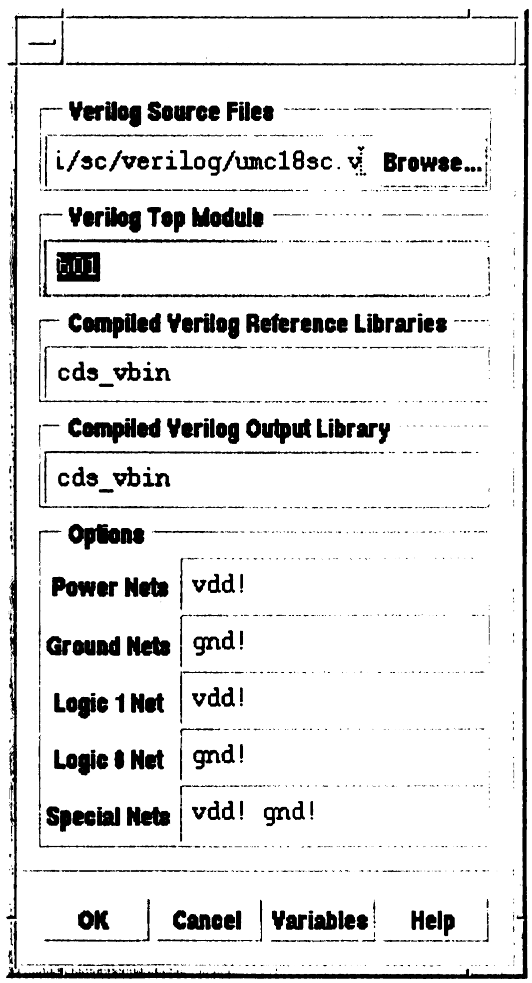

Figure 3.6: Import Verilog Dialogue Box

8) Initialize Floorplan (Figure 3.7).

a. Select *Floorplan > Initialize Floorplan.*

b. **I/O To Core Distance Left/Right:** 15 microns

c. **I/O To Core Distance Top/Bottom:** 15 microns

d. **Die Size Constraint Aspect Ratio:** 1.2

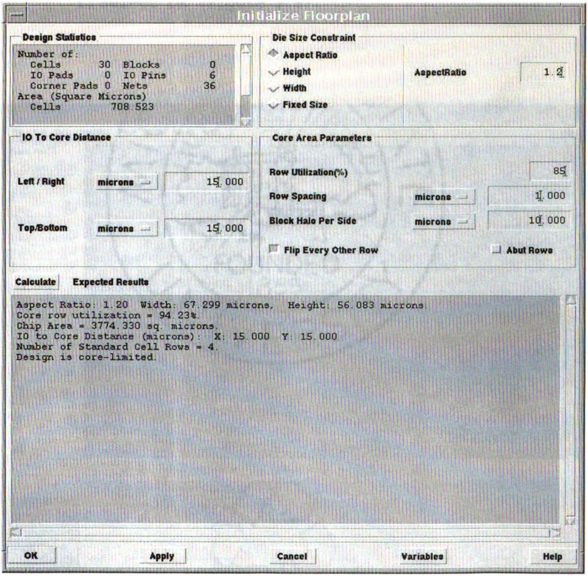

Figure 3.7: Initialize Floorplan Dialogue Box

e. **Core Area Parameters Row Spacing:** 1 micron

f. **Core Area Parameters Block Halo Per Side:** 10 microns

g. **Core Area Parameters**: Select *Flip Every Other Row.*

h. Select *Calculate.*

i. Select *OK* (Figure 3.8).

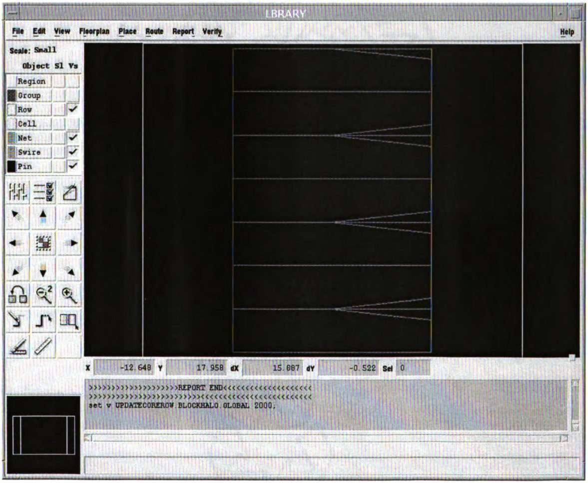

Figure 3.8: Floorplan

9) Place I/Os.

a. Select *Place > IOs…*

b. Select *OK*.

10) Place Cells.

a. Select *Place->Cells*.

b. Select *OK* (Figure 3.9).

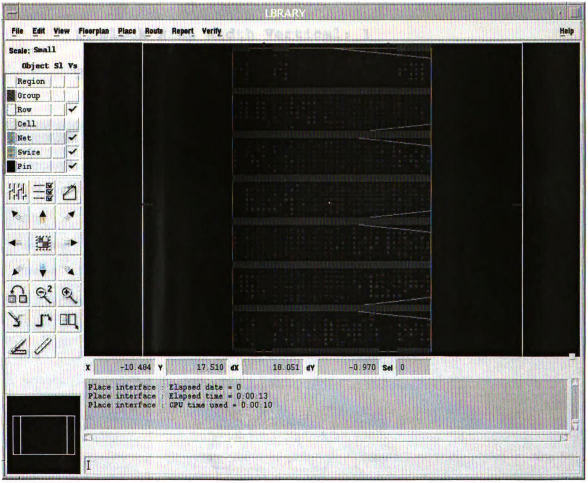

Figure 3.9: Placement of Cells

11) Refine I/O Placement.

a. Select *Place > IOs…*

b. **Placement Mode:** Select *Refine Pin Placement.*

c. Select *OK.*

12) Add Power Rings (Figure 3.10).

a. Select *Route > Plan Power.*

b. Select *Add Rings.*

c. **Core Ring Width Horizontal:** 1

d. **Core Ring Width Vertical:** 1

e. **Block Ring Width Horizontal:** 1

f. **Block Ring Width Vertical:** 1

g. Select *OK.*

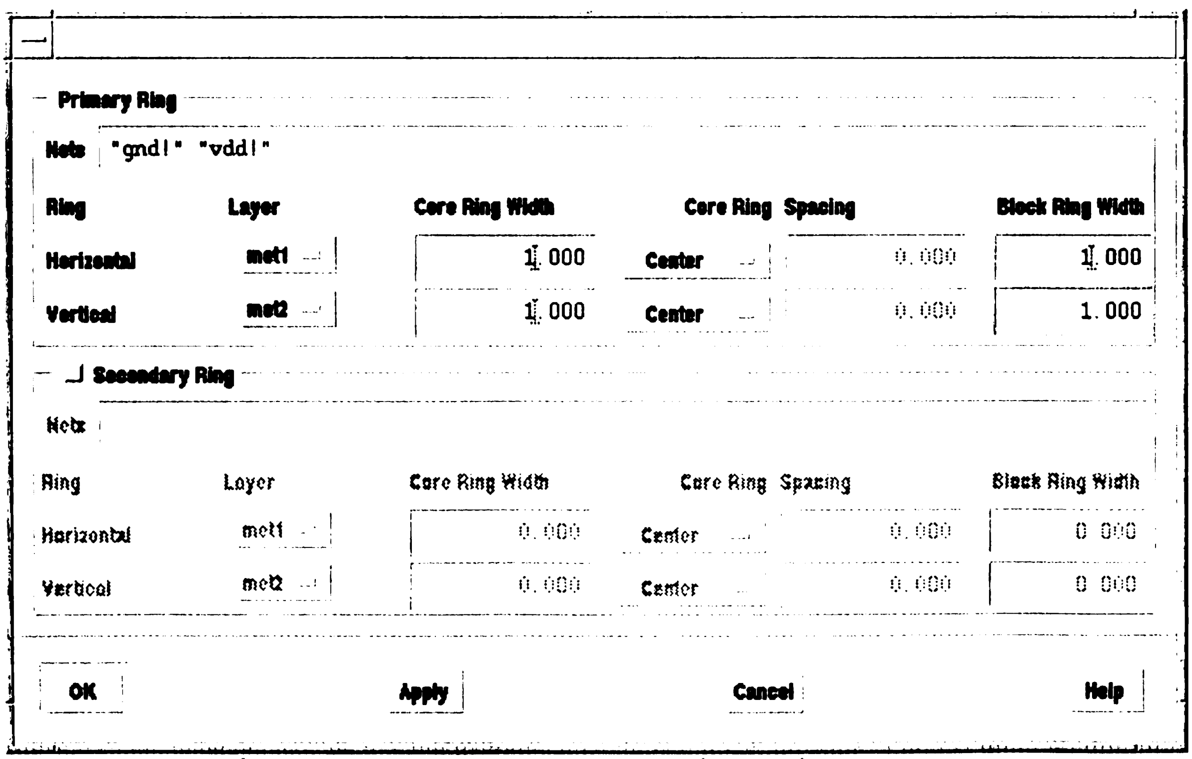

Figure 3.10: Plan Power Add Rings Dialogue Box

h. Select *Close*.

13) Route Power.

a. Select *Route > Connect Ring*.

b. Select *OK*.

14) Global Route.

a. Select *Route > Global Route*.

b. Select *OK*.

15) Final Route (Figure 3.11).

a. Select *Route > Final Route*.

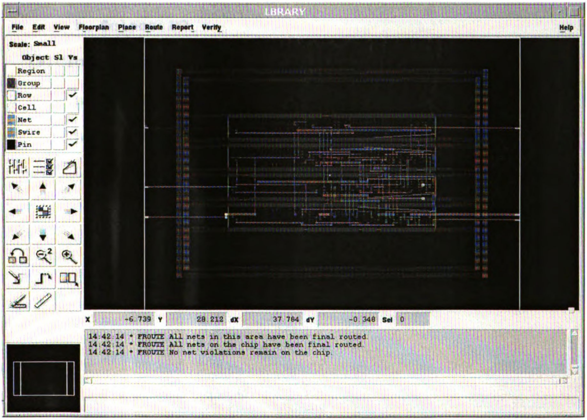

Figure 3.11: Final Routing of Chip

b. Select **Auto Search and Repair**

c. Select *OK*.

16) Verify Geometry.

a. Select *Verify* > *Geometry*.

b. Select *OK*.

17) Export DEF File.

a. Select *File* > *Export* > *DEF*.

b. **DEF File Name:** b01.def

c. Select *OK*.

18) Exit Silicon Ensemble®.

a. Select *File* > *Exit*.

b. Select *NO*.

19) Close terminal window.

a. >exit **RTN** *Exit the t shell.

b. >exit **RTN** *Exit the terminal.

3.4.3 *Virtuoso®*

This section describes how to launch Virtuoso®, create a new library with the correct technology, create a new cellview in the library, import the DEF file generated from Section 3.4.2, and export a GDSII file.

1) Launch a new terminal window.
    a. Right-click on desktop.
    b. Select *Utilities > Terminal*.
2) Load the t shell.
    a. >tcsh **RTN**
3) Change to correct directory.
    a. >cd tutorial **RTN**
4) Launch Virtuoso® (Figure 3.12).
    a. >source $SOFT/ic445 **RTN**
    b. >icfb *The window that has appeared is called the Command Interpreter Window (CIW). This is the main window from which other programs are launched.
5) Create a new library.
    a. Select *File > New > Library*.
    b. **Name:** tutorial
    c. **Compile a New Techfile**: Selected.
    d. Select *OK*.

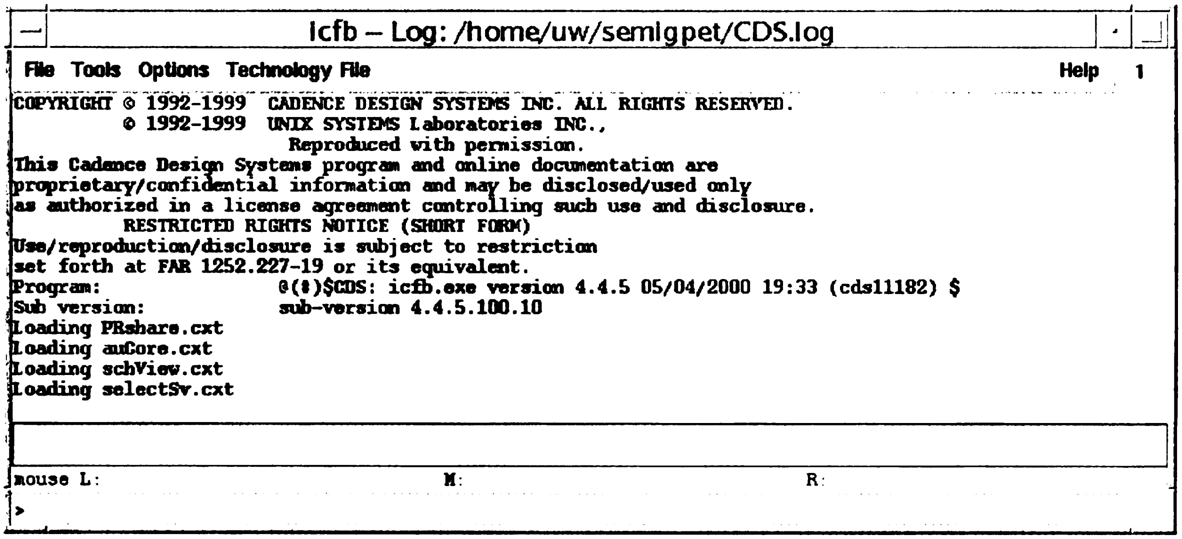

Figure 3.12: Command Interpreter Window (CIW)

e. **ASCII Technology File:** umc18.tf

f. Select *OK*.

6) Create new cellview (Figure 3.13).

a. Select *File > New > Cellview*.

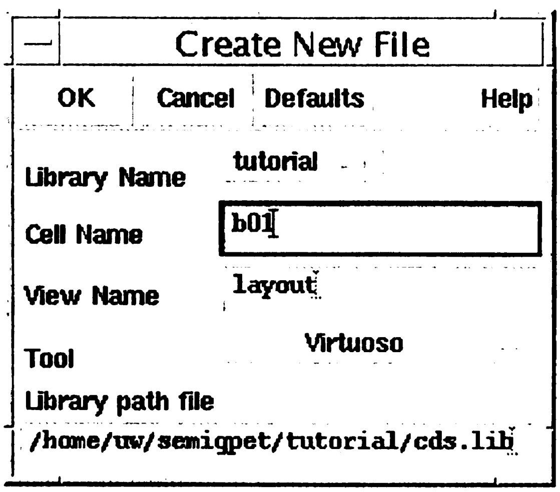

Figure 3.13: Create New File Dialogue Box

b. **Library Name:** tutorial

c. **Cell Name:** b01

d. **Tool:** Virtuoso

e. Select *OK.* *If a message appears concerning undefined packets, select *Yes*. The window that appears is called the Layout Editor.

7) Import DEF file.

a. In the CIW, select *File* > *Import* > *DEF*.

b. **DEF File Name:** b01.def

c. Select *OK.*

8) View entire design.

a. In the Layout Editor, select *Window* > *Fit All* (Figure 3.14). *This is the IC we synthesized in NaviGates® and placed and routed in Silicon Ensemble®. Notice that all lines are yellow. This is because we do not have the actual cell layouts in the Artisan library. We also do not have the display.drf file, which specifies layer colors and patterns. Also notice that there are many errors in the CIW. These can be ignored due to the same reasons.

9) Save Design.

a. Select *Design* > *Save*.

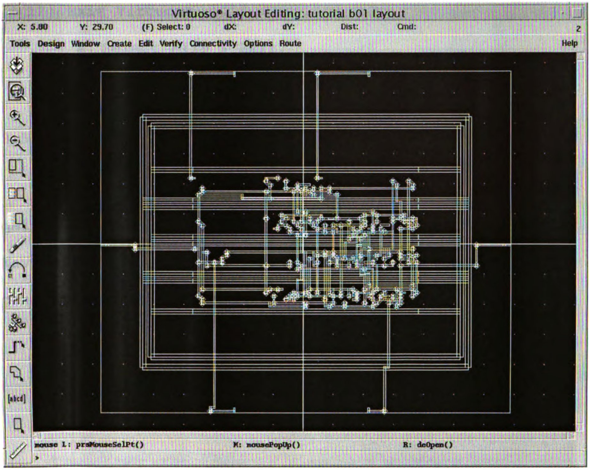

Figure 3.14: Final Chip Layout

10) Close the Layout Editor.

a. Minus sign in the upper-left corner of the Layout Editor window. *You cannot close the LSW.

11) Export GDSII file.

a. In the CIW, select *File > Export > Stream*.

b. **Template File**: template.streamout

c. Select *Load*.

d. Select *OK*. *Some warnings will appear, but these can be ignored.

12) Exit Virtuoso.

   a. Select *File* > *Exit*.

   b. Select *Yes*, if applicable.

3.5 *Summary*

After completing this tutorial, the reader should have gained an understanding of synthesis with Ambit® NaviGates®, P&R with Silicon Ensemble®, and GDSII file generation with Virtuoso®.

CHAPTER 4

LESSONS LEARNED

> *A successful person is one who can lay a firm foundation with the bricks that others throw at him or her.*
>
> -David Brinkley

While working on this thesis project, many lessons have been learned. Most have been the results of transcending difficulties. Finding a quality standard cell library, making contacts at other universities, and staying informed of changes made to the Cadence® operating environment are just three of the difficulties whose presence provided opportunities for learning.

The first step taken in finding a standard cell library for use with the Cadence® tools was an exhaustive search on the Internet. A few libraries and design kits were found, but not of the appropriate quality. Companies were contacted via telephone; promises were made; promises were never fulfilled.

It was not until a trip to Design Automation Conference in June of 2000 (DAC2000) that the quest for a standard cell library took shape. It was at DAC2000 where many contacts were made with library vendors, foundry representatives, university specialists, and Cadence® personnel.

After returning to Michigan, working with these contacts via telephone was much easier. The ability to attach a name with a face gave greater importance to our discussions.

A trip to Purdue University in May of 2000 provided invaluable contacts and a good understanding of MSU's place in VLSI education. It was at Purdue that I met Dr. Mark Johnson and Mr. Shawn Davidson. Mr. Davidson gave a tour of the Purdue VLSI design laboratory, demonstrated their Cadence® design flow, gave the Internet address where I could obtain the CMU SCL, and answered my Cadence® configuration questions. Dr. Johnson proved to be key in helping resolve issues around importing DEF files into Virtuoso®. He was the initial contact for resolving Cadence® issues. Dr. Johnson had already addressed most of the issues that were encountered while determining the cell-based design flow.

On a couple occasions throughout the past two years, updates to the operating environment and the Cadence® tools have had an adverse affect on the completion of this thesis. One example deals with the upgrade from IC 4.4.3 to IC 4.4.5. In December 2000, Cadence® sent DECS the newest release of the IC tool set. This upgrade generated many errors when using the EP SCL, which was originally intended for IC 4.4.3. This upgrade resulted in a major setback, for a new SCL compatible with IC 4.4.5 was needed.

Keeping abreast of changes to the operating environment and the Cadence® tools, making contacts at other universities, and traveling to conferences where contacts with company representatives can be made are all extremely important to the success of the VLSI design program at MSU.

CHAPTER 5

CONCLUSION

5.1 *Conclusion*

The completion of this thesis has provided MSU with a good foundation to build our VLSI design program. The following is a list of accomplishments:

- Learned and documented the cell-based design of an IC in conjunction with a SCL.
- Obtained and installed four DKs and six SCLs.
- Developed a tutorial that demonstrates the design of an IC using the Cadence® tools. The tutorial demonstrates design synthesis with Ambit® NaviGates®, P&R with Silicon Ensemble®, and GDSII file export with Virtuoso®. MSU faculty, staff, and students can use this tutorial as a starting point for designing ICs.
- Developed application notes that discuss how to source and run the Cadence® tools and the files associated with synthesis, P&R, and layout.
- Established important contacts with library vendors, other universities, foundry representatives, and Cadence® personnel.

5.2 *Future Work*

Even though much has been accomplished, more needs to be done to bring MSU up to par with other university's VLSI design programs. The following is a list of items that MSU should address:

- Make contacts with AMI and obtain SCLs for fabrication with MOSIS. This would allow MSU students to experience both cell-based and full custom design with a technology that can be fabricated at MOSIS. The AMI regional sales manager is James Beaton (847-776-4500).
- Obtain updated EP SCLs. The purpose of this would be to gain more flexibility with regard to foundry and technology.
- Obtain Synopsys Design Compiler. Synopsys's Design Compiler is the premier synthesis tool used in industry. This would allow MSU to utilize more SCLs while delivering experience with a well-known synthesis tool to MSU VLSI design students.
- Obtain and analyze North Carolina State University (NCSU) design kit. There is a lot of documentation and support for the NCSU design kit. Other universities use this kit, but whether or not the kit can serve all of MSU's purposes is undetermined.

- Learn how to develop our own standard cells. The ability to develop our own standard cells is a key component to the success of the VLSI program here at MSU. We would gain thorough knowledge of the interaction between SCLs and the Cadence® tools. We would also have the ability to generate our own SCL if needed.
- Make more contacts with other universities. Possible candidates include University of Illinois at Chicago, NCSU, and Purdue University. More contacts would yield a greater sharing of knowledge. It would also provide for other points of view concerning solutions to problems and VLSI design education.

APPENDICES

APPENDIX A

RUNNING CADENCE® EDA TOOLS

A.1 *Purpose*

The purpose of this appendix is to describe the methods used to run and access help for the Cadence® EDA tools.

A.2 *Background*

The following Cadence® products are available in the Michigan State University's DECS UNIX laboratories:

- Ambit® NaviGates®
- Deep Submicron Silicon Ensemble (DSM SE 5.3)
- Integrated Circuit (IC 4.4.5)
- PCB Systems Division (PSD 13.6)
- IC Craftsman (ICC 5.0)
- Signal Processing WorkSystem (SPW 4.5)
- Logical Design & Verification (LDV 3.0)

Once a tool has been sourced in the UNIX environment, many programs become available (Table A1). In order to obtain help for a particular tool, type 'openbook &' in the terminal where the tool was sourced. Currently all tools have openbook help except ICC 5.0.

Table A1: Cadence® Tools and Their Programs

Tool	Programs
Ambit®	ac_shell
DSM SE 5.3	Envisia™ place-and-route, gate array Affirma™ timing analyzer for full-custom Envisia™ gate array place-and-route ultra Envisia™ place-and-route system with signal integrity
IC 4.4.5	Virtuoso® schematic composer netlister to A Virtuoso® layout editor Virtuoso® schematic option for layout Virtuoso® compactor Virtuoso® HSPICE interface Assura™ interactive layout vs. schematic Dracula® interactive debugging environment Virtuoso® Layout Synthesizer Virtuoso® schematic composer to design Assura™ RC network reducer option Dracula® pattern generation option Cadence® SKILL development environment Virtuoso® EDIF 200 reader Virtuoso® EDIF 300 connectivity reader/writer Virtuoso® EDIF 300 schematic reader/writer Virtuoso® STREAM interface Virtuoso® CIF reader Virtuoso® CIF writer Virtuoso® XL layout editor Cadence® design framework integrator Virtuoso® schematic composer VHDL interface Virtuoso® schematic composer Verilog® interface Affirma™ analog simulation PLI Affirma™ analog statistical analysis option Affirma™ analog corners analysis option Affirma™ analog circuit optimizer option Affirma™ mixed-signal simulation interface Affirma™ Cadence® SPICE Affirma™ analog circuit simulator

Table A1 (cont'd).

Tool	Programs
IC 4.4.5	Affirma™ Verilog®-A simulation option Affirma™ RF simulation option Affirma™ RF IC package modeler Affirma™ HSPICE interface Affirma™ mixed-signal back-annotation Virtuoso® schematic composer Affirma™ analog design environment Affirma™ substrate coupling analysis Affirma™ AMS distributed processing option Dracula® physical verification Assura™ Diva physical verification
PSD 13.6	FET1100: Concept HDL FET1101S: Concept-SCALD UNIX only PX3000: Concept HDL expert PX3100: SPECCTRAQuest™ SI expert PX3300: PCB mixed-signal expert PX3400: digital logic SI library PX3410: memory SI library PX3420: FPGA SI library PX3430: microprocessor SI library PX3500: PCB librarian expert PX3700: PCB design expert PX4000: advanced package engineer expert PX4100: advanced package designer expert
ICC 5.0	Virtuoso® custom placer Cadence® chip assembly router
SPW 4.5	CDMATK: Cierto™ wideband CDMA library SPWB01: Cierto™ SPW university bundle WLAN: Cierto™ wireless local area networks library
LDV 3.0	Affirma™ simulation analysis environment Affirma™ native compiled Verilog simulator Affirma™ native compiled VHDL simulator

A.3 *Procedure*

This procedure describes the method used for running the Cadence® EDA tools. It uses the conventions outlined in Table 3.2.

1) Logon to one of the DECS UNIX stations.

2) Launch a terminal window.

   a. Right-click on the desktop.

   b. Select *Utilities* > *Terminal*. *Use a separate terminal for each tool.

3) Load the t shell.

   a. >tcsh **RTN**

4) Source the appropriate tool.

   a. >*source $SOFT/X* **RTN** *'X' denotes the source command, which can be found in Table A2.

   b. >*Y* **RTN** *'Y' denotes the executable command, which can be found in Table A2.

Table A2: Tool Source and Executable Commands

Tool	Source Command	Executable Command
Ambit®	ambit	ac_shell -visual
DSM SE 5.3	dsmse53	seultra
IC 4.4.5	ic445	icfb, icca, layoutPlus, icde, icds, icms
LDV 3.0	ldv3	signalscan
SPW 4.5	spw45	spw
PSD 13.6	psd136	projmgr
ICC 5.0	icc5	sbtool.exe

APPENDIX B

AMBIT® NAVIGATES® FILES

B.1 *Purpose*

The purpose of this appendix is to provide a better understanding of the input and output files associated with Ambit® NaviGates®. This appendix will briefly discuss the following file types: TLF/CTLF, and CMD.

B.2 *Timing Library Format File*

A Timing Library Format (TLF) file contains the timing information for a library. A CTLF file is a compiled TLF file. Currently, Ambit® NaviGates® can import only CTLF files.

TLF files can be generated from Synopsys LIB files by using the "syn2tlf" utility. To compile a TLF file into a CTLF file, the "tlfc" utility is needed. Unfortunately, "tlfc" is not available in the latest release of the Cadence® tools.

Figure B1 is a portion of the TLF file for the CMU library's D flip-flop (DFF). It contains timing information such as rise, fall, setup, and hold times.

```
Cell(dff_1x
        Celltype(seq)
// MODEL DEFINITION
 Model ( td_CLK_to_Q_rise_non
  (Spline
 (load_axis 0.065925  0.119070  0.305810  0.662880  1.219300  2.000100  )
 (input_slew_axis 0.002000  0.005319  0.016968  0.039213  0.073871  0.122500 )
 Data  (
 ( 0.888880  0.938710  1.086100  1.338400  1.722000  2.259200 )
 ( 0.909430  0.959230  1.106600  1.358900  1.742500  2.279600 )
 ( 0.971040  1.020900  1.168300  1.420600  1.804200  2.341400 )
 ( 1.048500  1.098300  1.245700  1.498000  1.881600  2.418700 )
 ( 1.130500  1.180400  1.327800  1.580100  1.963600  2.500800 )
 ( 1.211600  1.261300  1.408700  1.661000  2.044600  2.581700 )
 )
 )
  )
 Model ( td_CLK_to_Q_fall_non
  (Spline
 (load_axis 0.065929  0.119080  0.305870  0.662880  1.219300  2.000067 )
 (input_slew_axis 0.002000  0.007472  0.026681  0.063364  0.120510  0.200700 )
 Data  (
 ( 0.592780  0.668910  0.860280  1.150900  1.580000  2.180900 )
      ( 0.613510  0.689580  0.880840  1.171400  1.600500  2.201500 )
      ( 0.674800  0.750780  0.941950  1.232500  1.661600  2.262500 )
      ( 0.751900  0.828000  1.019000  1.309500  1.738500  2.339500 )
      ( 0.834110  0.910270  1.101100  1.391300  1.820300  2.421300 )
      ( 0.915440  0.991860  1.182700  1.472600  1.901600  2.502400 )
 )
 )
  )
 Model ( ts_CLK_to_Q_fall_non
  (Spline
 (load_axis 0.065929  0.119080  0.305870  0.662880  1.219300  2.000067 )
 (input_slew_axis 0.002000  0.007472  0.026681  0.063364  0.120510  0.200700 )
 Data  (
  ( 0.125910  0.185690  0.355750  0.681800  1.226400  2.012700 )
      ( 0.125710  0.185580  0.355950  0.681780  1.226500  2.012700 )
      ( 0.126490  0.185810  0.355860  0.681720  1.226400  2.012700 )
      ( 0.127070  0.186140  0.355730  0.681620  1.226400  2.012700 )
      ( 0.127270  0.186650  0.355600  0.681480  1.226300  2.012700 )
      ( 0.128600  0.188300  0.355850  0.681370  1.226300  2.012700 )
 )
 )
  )
 Model ( ts_CLK_to_Q_rise_non
  (Spline
 (load_axis 0.065925  0.119070  0.305810  0.662880  1.219300  2.000100 )
 (input_slew_axis 0.002000  0.005319  0.016968  0.039213  0.073871  0.122500 )
 Data  (
 ( 0.132870  0.183310  0.354260  0.689760  1.230800  2.000000 )
      ( 0.133130  0.183490  0.354410  0.689790  1.230800  2.000000 )
```

Figure B1: Sample TLF Entry for a DFF in CMU SCL

```
      ( 0.132850  0.183340  0.354260  0.689760  1.230800  2.000100 )
      ( 0.133120  0.183480  0.354410  0.689790  1.230800  2.000100 )
      ( 0.132830  0.183430  0.354300  0.689790  1.230800  2.000100 )
      ( 0.133570  0.183790  0.354620  0.689850  1.230800  2.000100 )
)
)
 )
Model ( td_D_to_CLK_rise_setup
 (Spline
(load_axis 0.065698  2.000050 )
(input_slew_axis 0.065925  2.000050 )
Data  (
( 0.567420  0.399310 )
      ( 1.045100  0.853550 )
)
)
 )
Model ( td_D_to_CLK_fall_setup
 (Spline
(load_axis 0.067191  2.000050 )
(input_slew_axis 0.065929  2.000050 )
Data  (
( 0.265060  0.237590 )
      ( 0.574380  0.406250 )
)
)
 )
Model ( td_D_to_CLK_fall_hold
 (Spline
(load_axis 0.067192  2.000050 )
(input_slew_axis 0.065929  2.000000 )
Data  (
 ( -0.520560  -0.328990 )
      ( -0.974800  -0.806650 )
)
)
 )
Model ( td_D_to_CLK_rise_hold
 (Spline
(load_axis 0.065042  2.000050 )
(input_slew_axis 0.065931  2.000000  )
Data  (
 ( -0.218220  -0.167270 )
      ( -0.492340  -0.359370 )
)
)
 )
Timing_props (
 Volt_mult_propagation (rise(1.1) fall(1.2))
 Temp_mult_propagation (rise(1.1) fall(1.2))
 Volt_mult_transition (rise(1.1) fall(1.2))
 Temp_mult_transition (rise(1.1) fall(1.2))
)
```

Figure B1 (cont'd).

```
Pin(D Pintype(data) Pindir( input) Timing_Props(Pin_Cap(0.009 )))
Pin(CLK Pintype(data) Pindir( input) Timing_Props(Pin_Cap(0.00658 )))
Pin(Q Pintype(data) Pindir( output) Function(IQ))
Register(
 Input(D)
 Clock(CLK)
 Output(Q)
)
Path(CLK => Q 01 01 Delay(td_CLK_to_Q_rise_non) Slew(ts_CLK_to_Q_rise_non))
Path(CLK => Q 01 10 Delay(td_CLK_to_Q_fall_non) Slew(ts_CLK_to_Q_fall_non))

Setup(D => CLK 01 posEdge td_D_to_CLK_rise_setup )
Setup(D => CLK 10 posEdge td_D_to_CLK_fall_setup )

Hold(D => CLK 01 posEdge td_D_to_CLK_rise_hold )
Hold(D => CLK 10 posEdge td_D_to_CLK_fall_hold )
)
```

Figure B1 (cont'd).

B.3 *Command File*

A command file (CMD) file is used to automate repetitive commands.

Figure B2 is a sample CMD file, which was used in the Chapter 2 tutorial.

```
rm -rf ./mylib
rm -rf ./lpm
mkdir ./mylib
mkdir ./lpm
set_vhdl_library mylib ./mylib
set_vhdl_library lpm ./lpm
set_vhdl_library WORK mylib
set_global hdl_vhdl_environment synopsys
read_vhdl -library lpm lpm_pack.vhd
read_vhdl b01.vhd
do_build_generic -all
read_ctlf /opte/cds/artlib/aci/sc/tlf/typical/timing-com1c_tt_n_n.ctlf
set_global target_technology typical
do_optimize
write_verilog b01.v
```

Figure B2: Sample Ambit® NaviGates® CMD File

APPENDIX C

SILICON ENSEMBLE® FILES

C.1 *Purpose*

The purpose of this appendix is to provide a better understanding of the input and output files associated with Silicon Ensemble®. This appendix will cover the following file types: LEF, and DEF.

C.2 *Library Exchange Format File*

The Library Exchange Format (LEF) file contains process technology and cell data for a standard cell library in ASCII format. The information for a standard cell library may be contained in multiple LEF files (e.g. one for standard cells and one for I/Os). LEF files can be created manually or generated with tools such as Abstract.

The LEF file must contain all cells and ports in the TLF file, be of version 5.1 or higher, and have power and ground pins defined for each cell. More information on TLF files can be found in Appendix B.

C.3 *Design Exchange Format File*

The Design Exchange Format (DEF) file contains a netlist for the design, which is a list of all the cells and how they are connected. A DEF file also contains all the physical constraints for the design. DEF files can be created manually or generated with tools such as Preview.

APPENDIX D

VIRTUOSO® FILES

D.1 *Purpose*

The purpose of this appendix is to provide a better understanding of some of the input and output files associated with Virtuoso®. This appendix will cover the following file types: TF, and GDSII.

D.2 *Technology File*

A technology file (TF) contains information such as layer definitions, layer rules, physical rules, electrical rules, device definitions (e.g. contacts, pins), compactor rules, and place and route rules (Figure D1).

```
;**********************************
;
; LAYER DEFINITION
;**********************************
;
layerDefinitions(
 techLayers(
 ;( LayerName              Layer#    Abbreviation )
 ;( ---------              ------    ----------- )
 ;User-Defined Layers:
  ( nwell                   1       nwell       )
  ( diff                   2       diff        )
  ( poly                    3       poly        )
techDisplays(
 ;( LayerName   Purpose     Packet        Vis Sel Con2ChgLy DrgEnbl Valid )
 ;( ---------   -------     ------        --- --- --------- ------- ----- )
  ( nwell      drawing    creamthickLine2_L t t t t t )
  ( diff      drawing    redbackSlash_S    t t t t t )
  ( pplus      drawing    magenta          t t t t t )
  ( poly      drawing    bluecrossthickLine t t t t t )
 ;**********************************
 ;
; LAYER RULES
;**********************************
;
layerRules(
 streamLayers(
 ;( layer     streamNumber   dataType      translate )
 ;( -----     ------------   --------      --------- )
  ( ("nwell" "drawing")  2     0           t     )
  ( ("diff" "drawing")  1     0          t     )
;**********************************
;
; PHYSICAL RULES
;**********************************
;
physicalRules(
 spacingRules(
 ;( rule                  layer1       layer2       value    )
 ;( ----                  ------       ------       -----    )
  ( minSpacing              "met1"                     0.26    )
  ( minSpacing              "via1"                    0.28    )
 ;**********************************
 ;
; ELECTRICAL RULES
;**********************************
;
electricalRules(
 characterizationRules(
 ;( rule                  layer1       layer2       value    )
 ;( ----                  ------       ------       -----    )
  ( areaCap               "met1"                    0.0     )
  ( areaCap               "met2"                    0.0     )
 ;**********************************
 ;
; P&R RULES
;**********************************
;
prRules(
 prRoutingLayers(
 ;( layer                preferredDirection  )
 ;( -----                ------------------ )
  ( met1                 "horizontal" )
```

Figure D1: Selected Sections of a Technology File (TF)

D.3 *GDSII File*

The GDSII file is the physical description of your circuit. This is the file that is sent to IC fabrication services, such as MOSIS or EP.

REFERENCES

REFERENCES

[1] Semiconductor Industry Association. World Market Sales & Shares 1982-1990. [Online] Available http://www.semichips.org/stats/shares.htm, July 16, 2001.

[2] Semiconductor Industry Association. World Market Sales & Shares 1991-2000. [Online] Available http://www.semichips.org/stats/shares2.htm, July 16, 2001.

[3] Semiconductor Industry Association. Semiconductor Forecast Summary 2000-2003. [Online] Available http://www.semichips.org/stats/forecastsum_spring.pdf, July 16, 2001.

[4] Tutorials found on the Internet (e.g. Purdue University, Iowa State University, Canadian Microelectronics Corporation) and in Cadence® help documentation (Openbook) were referenced during the compilation of the tutorial.

[5] Armstrong, James and F. Gail Gray. VHDL Design Representation and Synthesis. 2nd ed. Upper Saddle River, NJ: Prentice-Hall, 2000.

[6] Arnold, Mark Gordon. Verilog Digital Computer Design: Algorithms into Hardware. Upper Saddle River, NJ: Prentice-Hall, 1999.

[7] Anderson, Gail and Paul. The UNIX™ C Shell Field Guide. Englewood Cliffs, NJ: Prentice-Hall, 1986.

[8] Baker, R. Jacob, Harry W. Li and David E. Boyce. CMOS Circuit Design, Layout, and Simulation. IEEE Press Series on Microelectronic Systems. New York: IEEE Press, 1998.

[9] Ciletti, Michael D. Modeling, Synthesis, and Rapid Prototyping with the Verilog™ HDL. Upper Saddle River, NJ: Prentice-Hall, 1999.

[10] Glass, Grahm. UNIX® for Programmers and Users: A Complete Guide. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[11] Kang, Sung-Mo and Yusuf Leblebigi. CMOS Digital Integrated Circuits: Analysis and Design. 2nd ed. Boston: McGraw-Hill, 1999.

[12] Rabaey, Jan M. Digital Integrated Circuits: A Design Perspective. Prentice Hall Electronics and VLSI Series. Upper Saddle River, NJ: Prentice-Hall, 1996.

[13] Uyemura, John P. CMOS Logic Circuit Design. Boston: Kluwer Academic, 1999.

[14] Yalamanchili, Sudhakar. Introductory VHDL: From Simulation to Synthesis. Prentice Hall Xilinx Design Series. Upper Saddle River, NJ: Prentice-Hall, 2001.