DATA ACQUISITION SYSTEM BASED ON FPGA

Oleh

Nuril Syahira binti Mahrus

Disertasi ini dikemukakan kepada
UNIVERSITI SAINS MALAYSIA

Sebagai memenuhi sebahagian daripada syarat keperluan
untuk ijazah dengan kepujian

SARJANA MUDA KEJURUTERAAN (KEJURUTERAAN ELEKTRONIK)

Pusat Pengajian Kejuruteraan
Elektrik dan Elektronik
Universiti Sains Malaysia **Mac 2005**

ABSTRAK

Sebagai memenuhi syarat pengijazahan, satu sistem perolehan data berdasarkan FPGA telah dibina. Data dalam bentuk analog, yang diperoleh dari pembekal kuasa (isyarat dalam bentuk voltan) akan ditukarkan kepada bentuk digital dengan menggunakan ADC0809. Sistem ini dikawal sepenuhnya oleh FPGA dengan menggunakan perisian *Xilinx Foundation Series 2.1*, melalui *Project Manager*. Bahasa yang digunakan untuk membina aturcara ialah *VHDL code*. Peranti Xilinx yang digunakan adalah XC4010 PC84. Data yang diperoleh dari ADC0809 (dalam bentuk digital) akan diproses untuk membolehkannya dipaparkan pada LCD (dengan cara menukarkan kod Hex kepada kod ASCII). Selepas data ini telah ditukarkan kepada bentuk kod ASCII, data ini akan dipaparkan pada LCD. Pada masa yang sama, data ini juga akan dibandingkan dengan pembolehubah yang telah ditetapkan, iaitu 3F(dalam Hex), yakni bernilai 1.24V. Jika data yang diperoleh adalah melebihi pembolehubah tersebut, maka LED akan menyala. LED tidak akan menyala jika keadaan sebaliknya terjadi. Ini direka untuk menunjukkan sistem pengawasan.

ABSTRACT

For undergraduate project, a data acquisition system based on FPGA has been developed. An analog data, which comes from power supply (in voltage signals) is gathered and then be converted to digital signals. This is done by ADC0809. This system is fully controlled by FPGA using the Xilinx Foundation Series 2.1, by Project Manager. The device used is XC4010 PC84 and the language used to develop the coding is VHDL code. Data from the ADC0809 (which is in digital form) will be processing to make sure that it can appear on LCD. To do this, the data first must be converted from the Hex code to the ASCII code. After the data has been converted to ASCII code, then it will be displayed on the LCD. At the same time, the data will be compared to a fix variable, which is 3F in Hex and equal to 1.24V. If the data is beyond the variable then LED will turning ON, and if the data is less than that (the variable), then the LCD will not turning ON. It is designed to show the monitoring system.

ACKNOLEDGEMENTS

Alhamdullillah... In the name of Allah, most grates and most merciful.

First of all, I would like to take this opportunity to express my deepest gratitude and appreciation to Puan Zaini, my final year project supervisor. She had been very supportive for me to continue my project. More importantly, her willingness to advice and guide me along throughout this period. She is too patience to guide a new FPGA user like me. Thank you very much.

I would like to extend my appreciation to all my friends who did the FPGA for the Final Year Project too, because they also help me a lot to use this system.

Not to forget, to Encik Khairul and Puan Shahida, the technicians of Microprocessor Laboratory, the place where I did my project. They are agreed to borrow me the instrument in the laboratory.

Last but not least, to Dr.Othman Sidek, the Electrical and Electronics Dean as well as my second examiner who had express his great interest in my project and was willing to be my second marker.

Thank You.

TABLE OF CONTENTS

REFERENCES

APPENDIX A: DATA SHEET

1. XC 4000X
2. ADC 0809
3. LCD

APPENDIX B: HDL CODE

1. ADC
2. UPPER
3. LOWER
4. MUX
5. LCD

APPENDIX C: SCHEMATIC EDITOR

1. COMBINATIONAL BLOCKS
2. COMPARATOR SCHEMATIC (MACRO)

LIST OF FIGURES AND TABLES

Page

CHAPTER 1: INTRODUCTION

1.1 Project Overview

Generally, Data Acquisition (DAQ) System consists of amplifier, multiplexer, analog to digital converter (ADC), controller to control the inputs data and a screen monitor. For this under graduate final year project, one DAQ System based on FPGA is designed. FPGA is stands for Field Programmable Gate Arrays. FPGA is chose because FPGA is the high density ASIC that allow to re-configurable the hardware without change its physical structure. It also implement the system to do the operation for the millions logic gate. Designing based on FPGA can be done in a short time because they are a few process that can be leaved such as fabrication and casing.

Data are first captured and subsequently translated into usable signals using transducers. In this discussion, usable signals are assumed to be electrical voltages. When there is more than one analog input, they are subsequently sent to an analog multiplexer (MUX). The analog signal is then conditioned using signal conditioners. There are two additional steps for those conditioned analog signals. First they must be sampled and next converted to digital data. This conversion is done by analog-to-digital converters (ADC).

Many applications require information capturing from more than one channel. The use of MUX is to cater to multiple inputs. When the MUX is addressed to select an input, say x_i (t), the same address will be decoded by the decoding logic to generate another address, which is used in addressing the programmable register. The programmable register contains further information regarding how to handle x_i (t). The outcome of the register is then used in subsequent tuning of the signal conditioner. The MUX address generator could be programmed in many ways; one of the simplest ways is to scan input channels in a cyclic fashion where the address can be generated by means of a binary counter. FPGA is used in addressing MUX.

FPGA is also used to get data from the ADC and sent to LCD for displaying the result through the ASCII codes. The system which is hex to ASCII codes needs to be implementing to make sure that the LCD can display the right output or number that need.

The controlling device, which is Xilinx (XC 4010 PC 84) in this case, first selects the desired input channel. To do this, a 3-bit channel address is placed on the A,

B, C input pins, and the ALE input is pulsed positively, clocking the address into the multiplexer address register. To begin the conversion, the START pin is pulsed. On the rising edge of this pulse, the internal registers are cleared and on the falling edge the start conversion is initiated. The address A, B, C pins, ALE pins and Start pin as well as the OE pins are the output from XC 4010 PC 84 to ADC 0809. When EOC, which is the input from the ADC 0809, goes high, it means it is the end of conversion and the data resulting from the conversion is ready to be read.

After the data has been constructed, it will then be compared to a fix variable. This comparison is needed to build a monitoring system in the design. If the system monitored that the constructed data is exceeding the fix variable, then LED will turning ON.

Figure 1.1 shows the flow chart describing the easiest way how the system works.

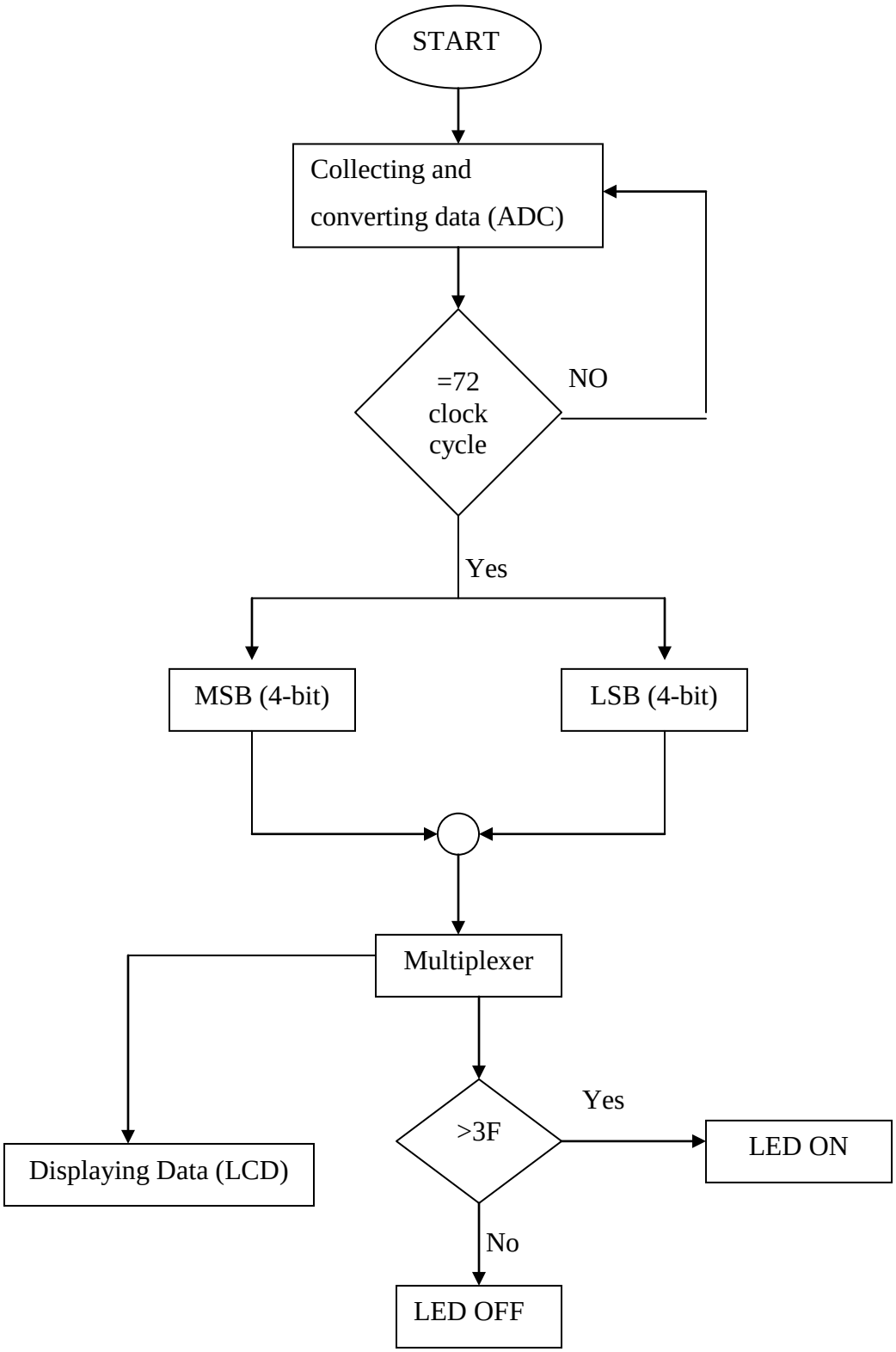

Figure 1.1: Flow Chart of the System

1.2 Objectives and Scope of Project

This monitoring system usually used by an aero plane, to indicate the navigation, the fuel, and many more. Besides the aero plane, this monitoring system also can be applied to indicate the temperature in oven or in a room that need a specific temperature. It will sense any changes that occurred and then display it and will blinking or honking if it comes to a dangerous level.

This project related to design a monitoring system which is used the data acquisition system based on FPGA. To achieve this target, two main objectives are discovered:

1. Construct a data acquisition system based on FPGA
2. Design a monitoring system which is used the data acquisition system.

The first objective, which is to construct a data acquisition system based on FPGA will be done by gathering the usable signals, which will be electrical voltages, in ADC0809 and will be converted to digital signals. Then these signals will then go through a controlling device, which is Xilinx (XC 4010 PC 84). After that, the data then will be display on the LCD.

LED (light-emitting diode) will be used to indicate whether the data (inputs), which have been constructed before, exceed or not the variable that has been fixed in a design of this system. If the data is exceed the fixed variable, the LED then will turning ON. This approach is for achieving the second objective of the design of this system, which is to design a monitoring system.

The monitoring system is very useful to monitor whether the system is in a dangerous level or not.

1.3 Structure of Report

Chapter 2: A survey on the history of FPGA, what is data acquisition is all about and the reason why FPGA has been chosen.

Chapter 3: The step by step to implement the FPGA

Chapter 4: The result and the explanation of each block (simulation), hardware developing (connections between the hard wares), and the final result (finishing output).

Chapter 5: The conclusion of the projects as well as the future approach to improve this project.

CHAPTER 2: LITERATURE SURVEY

2.1 Field Programmable Gate Array (FPGA)

2.1.1 History of FPGA

By the early 1980s there were many companies providing programmable logic devices, each which its own unique set of architectural features. As these devices become more complex and varied, the need grew for higher levels of design abstraction and for automated conversion of higher-design descriptions to programmable logic implementations. Simulation tools also became increasingly important; the size and complexity of programmable logic applications were now growing to the point where testing based on device inputs and outputs were not sufficient.

Although the use of programmable logic devices for creating logic circuits grew dramatically in the early to mid 1980s, they did not take hold among embedded systems designers and programmers due to their relatively small size and the wide availability and low cost of standard microcontrollers. While microcontrollers were (and remain) exceptionally slow in terms of data throughput relative to what is possible using programmable logic, the tremendous flexibility provided by microcontrollers and micro processors, along with their ability to host much larger algorithms and applications made them the obvious choice for all but he simplest and most performance-critical functions.

Three factors combined in the mid 1980s to cause a change in the way that programmable logic was applied, and in its applicability to hardware acceleration of software computations. These factors were reprogram ability, more flexibility interconnect architectures and tools for the automated synthesis of circuitry from standard hardware description language.

The new, more flexible device architectures represented by FPGA devices (which were produced by Xilinx, Inc., Altera Corporation, and others) were also critical. These devices had dramatically higher logic capacities and more general-purpose features than had existed previously, and therefore provided a new set for performing complex computations directly in hardware.

Finally the emergence of more powerful logic synthesis software tools and the development and standardization of two powerful hardware description languages, VHDL and Verilog, made it possible for application developers to work at a higher level

of abstraction, leaving to the design tools the details of actual implementation of an algorithm as low-level logic gates, registers and the like.

By the start of 1990s, it was clear that programmable logic devices had matured to the point where the high-performance software algorithms and even complete applications could be implemented in hardware.

At the turn of century, the primary use of FPGAs was still almost overwhelmingly for traditional hardware functions; controllers, interface logic and peripheral integration. FPGA based computing had not yet caught hold.

In recent years, two important developments have combined with ever-increasing device densities to make FPGA-based computing practical for mainstream applications. The first of these developments have been the availability of full-featured embedded FPGA processors cores, including the Micro Blaze ad Power PC cores available from Altera, and other processors cores available from third –party IP suppliers. [1]

2.1.2 Why FPGA Has Been Chosen

The basic idea underlying the architecture of an FPGA is very simple. In general, combinational and sequential circuits can be implemented directly in silicon. Such ASICs produce the highest performance but can perform only one function, and it is not that practical because it may needs high cost. [2]

FPGAs feature a gate-array-like architecture has three basic structures; configuration logic blocks (CLBs), I/O cells and programmable switching matrix. It has a matrix logic cells surrounded by a periphery of I/O cells. Segments of metal interconnections can be linked in an arbitrary manner by programmable switches to form the desired signals nets between the cells. This choice allows the user to both design and programmed a device on premises; thus resulting in short design and production lead times.

These features make the FPGA devices viable for small production runs and the natural choice of technology where frequent production variations are required to meet specific customer needs. Furthermore, integrating the complete system on to the chip saves on both space and assembly costs and leads to an improvement in reliability and design accuracy. [3]

2.2 Data Acquisition System (DAQ)

Data acquisition includes everything from gathering data, to transporting it and to storing it. A data acquisition phase involves a real-time computing environment where the computer must be keyed to the time scale of the process.

Today, the data acquisition system (DAQ) are more widely used in many fields, such as engine diagnosis, safeguards applications, remote sensing, clinical monitoring, power quality analysis and many more. This large spectrum of applications requires higher performance system. This performance is expressed in terms of not only high sampling frequency, high resolution but also real-time compression of acquired samples.

The proposed DAQ adopt an approach including timing information directly in the data stream, using a compact coding format, with consequent data reduction. These systems are realized with one using an analog architecture and one using a digital architecture.

The DAQ based on analog architecture, utilizes a voltage control oscillator (VCO) to generate a signal. This signal controls analog-to-digital converter to sample the input signal. [4]

2.3 Introduction to System's Instrument

2.3.1 XC4010 PC84

The XC 4010 PC84 board is in XC 4000 XL series. The biggest advantages of this device are significantly increased system speed, greater capacity and new architectural features, particularly Select-RAM memory. The XC 4010 PC 84 also offer many new routing features, including special high-speed clock buffers that can be used to capture input data in minimal delay. The features of this device are:

- Offers a high performance with maximum 3.3 V
- Has high capacity with over 180 000 usable gates.
- It I/Os are tolerant with 5V
- Buffered are interconnect for maximum speed blocks.
- It has optional multiplexer or 2-input function generator on device output.

This XC 4000 Series devices are implemented with regular, flexible, programmable architecture of Configurable Logic Blocks (CLB), interconnected by a powerful hierarchy of versatile routing resources and surrounded by perimeter of Input/Output Blocks (IOBs). This device has generous routing resources to accommodate the most complex interconnect pattern.

The devices are customized by loading configuration data into internal memory cells. The FPGA can either actively reads its configuration data from an external serial or byte-parallel PROM (master mode), or the configuration data can be written into the FPGA from an external device (slave and peripheral modes).

XC 4000 Series FPGAs are supported by powerful and sophisticated software, covering every aspect of design from schematic or behavioral entry, floor planning, simulation, automatic block placement and routing of interconnects, to the creation, downloading and readback of the configuration bit stream.

Because of Xilinx FPGAs can be reprogrammed an unlimited number of times, they can be used in innovative designs where hardware is changed dynamically, or where hardware must be adapted to different user applications. [5]

For downloading the software that has been built, EPROM 27C512 is used. The chip is an UV-erasable EPROM (UV-EPROM) type. This chip has a window that is used to shine ultraviolet (UV) radiation to erase it contents. It is located in the socket on the XC 4010 PC84 board.[6]

Figure 2.1 shows the XC 4010 PC84 board.

Figure 2.1: XC 4010 PC84

2.3.2 ADC 0809

The ADC 0809 Data Acquisition Device implement on a single chip most the elements of the standard data acquisition system. It contains an 8-bit A/D converter, 8-channel multiplexer with an address input latch, and associated control logic. Following are the features of the ADC 0809:

- Easy interface to all microprocessors
- Operates ratio metrically or with 5Vdc or analog span adjusted voltage reference
- No zero or full scale adjust required
- 8-channel multiplexer with address logic
- 0V to 5V input range with single 5V power supply
- Outputs meet TTL voltage level specification

Figure 2.2 shows the ADC 0809.

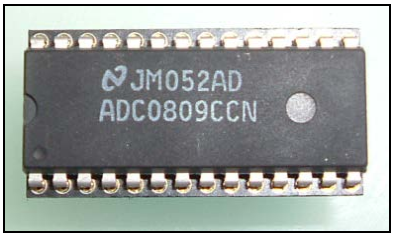

Figure 2.2: ADC 0809

Figure 2.3 shows the layout pin of ADC0809.

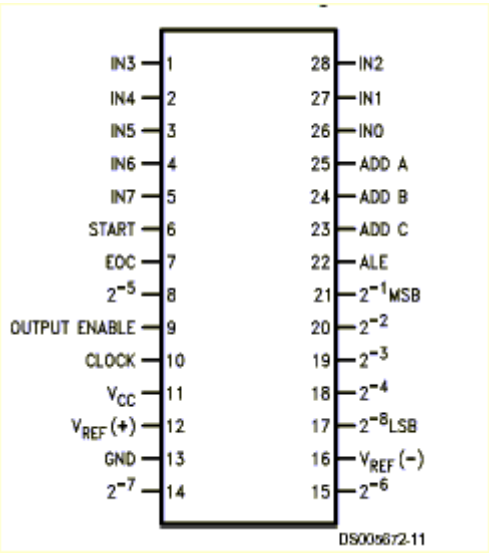

Figure 2.3: Layout Pin ADC0809

The operation of these converters by a microprocessor or some control logic is very simple. The controlling device first selects the desired input channel. A 3-bit channel address is placed on the A, B, C input pins, and the ALE input is pulsed positively, clocking the address into the multiplexer address register. To begin the conversion, the START pin is pulsed. On the rising edge of this pulse the internal register are cleared and on the falling edge the start conversion is initiated.

There are 8 clock periods per approximation. A START pulse can occur any time during this cycle but the conversion will not actually begin until the converter internally cycles to the beginning of the next 8 clock periods sequence. As long as the START pin is held high, no conversion begins, but when the START pulse is taken low, the conversion will start within the 8 clock periods.

The EOC output is triggered on the raising edge of the START pulse too. It is also controlled by the 8 clock period cycle, so it will go low within 8 clock periods of the rising edge of the START pulse. Once EOC does go high, this signal the interface logic that data resulting from the conversion is ready to be read. Figure 2.4 shows the timing diagram of ADC 0809. [7]

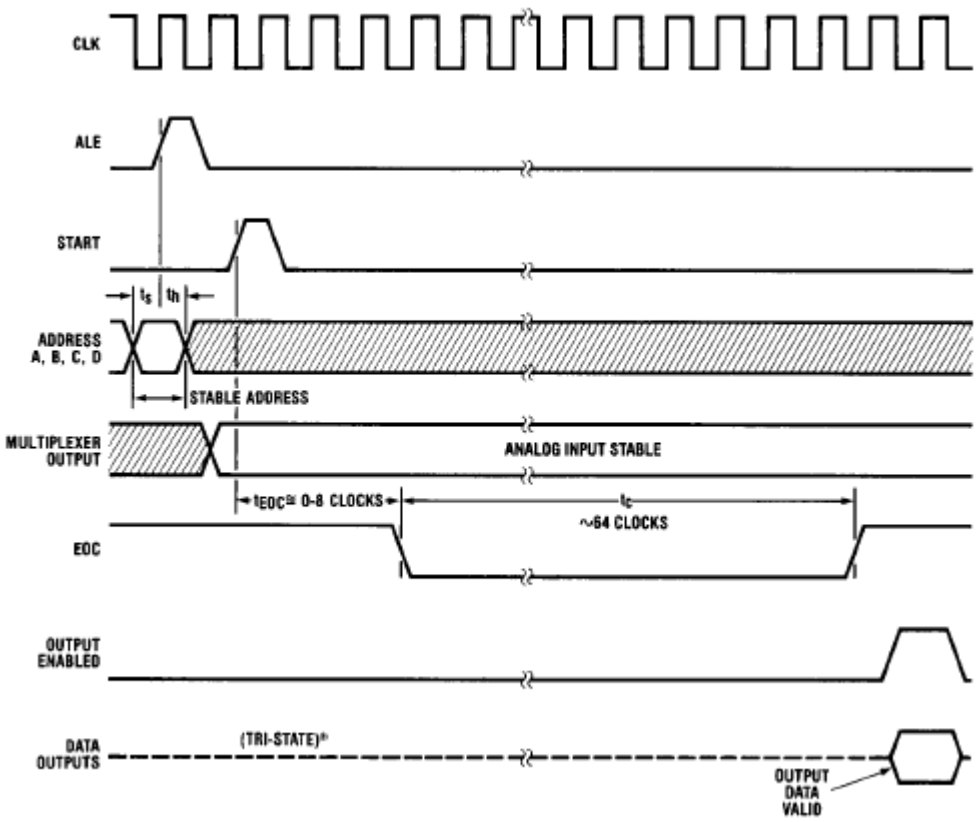

Figure 2.4: ADC0809 Timing Diagram

2.3.3 Liquid Crystal Diode (LCD) 16 x 2 Display

In recent years, the LCD display is finding widespread use replacing LEDs (seven-segment LEDs or multi-segment LEDs). This is due to the following reasons:

- The declining prices of LCDs.
- The ability to display numbers, characters and graphics.
- Incorporation of a refreshing controller into the LCD, thereby relieving the CPU of the task of refreshing the LCD.
- Ease of programming for characters and graphics. [6]

Figure 2.5 shows the LCD 16 x 2 displays.

Figure 2.5: LCD 16 x 2

The LCD 16 characters x 2 lines have 14 pins in used. The function of each pin is given in Table 2.1.

Table 2.1: Pin Description for LCD

Pin	Symbol	I/O	Description
1	Vss	Power	Ground
2	Vcc	Power	+5V power supply
3	Vo	Analog	Variable resistor (to control contrast)
4	RS	Input	RS=0 (select

			command register) RS=1 (select data register)
5	RS	Input	RW=0 (for write) RW=1 (for read)
6	E	I/O	Enable
7	D_0	I/O	8-bit data bus
8	D_1	I/O	8-bit data bus
9	D_2	I/O	8-bit data bus
10	D_3	I/O	8-bit data bus
11	D_4	I/O	8-bit data bus
12	D_5	I/O	8-bit data bus
13	D_6	I/O	8-bit data bus
14	D_7	I/O	8-bit data bus

Figure 2.6 shows the timing diagram for LCD.

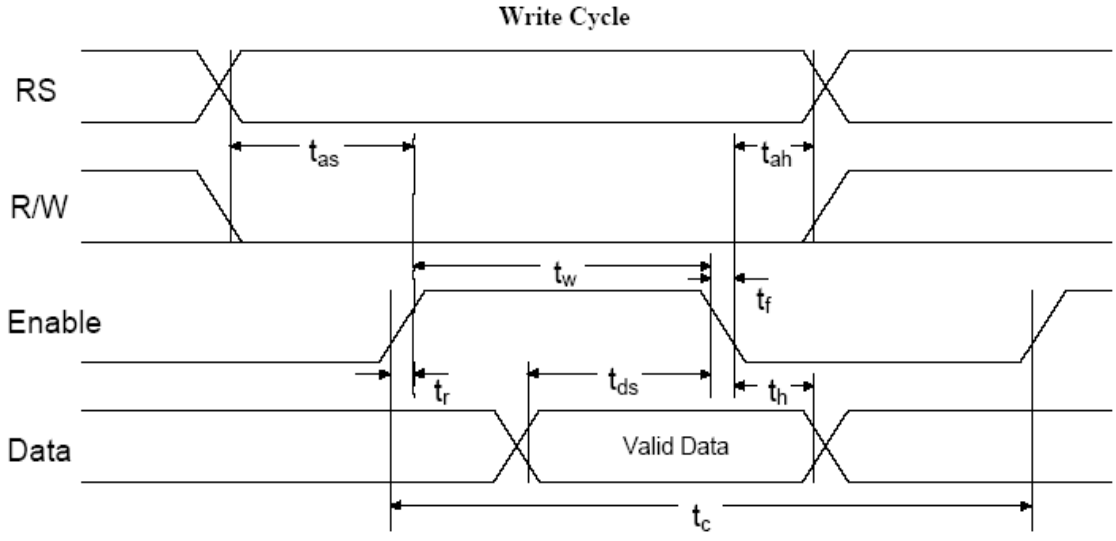

Figure 2.6: LCD Timing Diagram

To make sure that LCD can display the right output or number needed hex-to-ASCII code is need to be implementing. ASCII is stands for American Standard Code

for Information Interchange. The ASCII code is used to transfer of alphanumeric information between a computer and external devices such as a printer or another computer. Alphanumeric codes are codes that represent numbers and alphabetic characters (letters). Table 2.2 shows the partial listing of ASCII code.

Table 2.2: Partial listing of ASCII code

Character	ASCII	Hex	Character	ASCII	Hex
A	0100 0001	41	g	0110 0111	67
B	0100 0010	42	h	0110 1000	68
C	0100 0011	43	i	0110 1001	69
D	0100 0100	44	j	0110 1010	6A
E	0100 0101	45	k	0110 1011	6B
F	0100 0110	46	l	0110 1100	6C
G	0100 0111	47	m	0110 1101	6D
H	0100 1000	48	n	0110 1110	6E
I	0100 1001	49	o	0110 1111	6F
J	0100 1010	4A	p	0111 0000	70
K	0100 1011	4B	q	0111 0001	71
L	0100 1100	4C	r	0111 0010	72
M	0100 1101	4D	s	0111 0011	73
N	0100 1110	4E	t	0111 0100	74
O	0100 1111	4F	u	0111 0101	75
P	0101 0000	50	v	0111 0110	76
Q	0101 0001	51	w	0111 0111	77
R	0101 0010	52	x	0111 1000	78
S	0101 0011	53	y	0111 1001	79
T	0101 0100	54	z	0111 1010	7A
U	0101 0101	55	0	0011 0000	30
V	0101 0110	56	1	0011 0001	31
W	0101 0111	57	2	0011 0010	32
X	0101 1000	58	3	0011 0011	33
Y	0101 1001	59	4	0011 0100	34

Z	0101 1010	5A	5	0011 0101	35
a	0110 0001	61	6	0011 0110	36
b	0110 0010	62	7	0011 0111	37
c	0110 0011	63	8	0011 1000	38
d	0110 0100	64	9	0011 1001	39
e		65	:	0011 1010	3A
f		66	blank	0010 0000	20

2.3.4 Light-Emitting Diode (LED)

LED is a complex semiconductor that converts an electrical current into light. The conversion process is fairly efficient in that it generates little heat compare to the incandescent light. LED is of interest for fiber optic because of five characteristics:

1. It is small.
2. It possesses high radiance.
3. The emitting area is small, comparable to the dimension of the optical fibers.
4. It has very long life, offering high reliability.
5. It can be modulated (turned off and on) at high speed.

2.3.5 DC Power Supply

The dc power supply is an indispensable instrument on any test bench. The power supply converts an ac power from the standard wall outlet into regulated dc voltage. All digital circuits require dc voltage to operate. The power supply is used when a new circuit is bread-boarded or when a PC board is pulled from a system for testing and is no longer operating from the internal system power supply. Figure 2.7 shows the DC power supply that used in the laboratory. [8]

Figure 2.7: DC Power Supply

2.3.6 Function Generator

The function generator is a signal source that provides pulse waveforms, as well as sine wave and triangular waveforms. Many function generators have logic-compatible outputs to provide proper level waveforms as inputs to digital circuits in order to check the operation. Figure 2.8 shows the function generator that used in laboratory. [8]

Figure 2.8: Function Generator

CHAPTER 3: FPGA IMPLEMENTATION

3.1 Project Manager

Project Manager is an application that manages and supervises al Foundation Series tools that involved in the design process, and it integrates all Foundation tools into a unified environment. This environment includes such tools as Schematic Editor, HDL Editor, State Diagram Editor, fast gate-level Logic Simulator and external third-party programs. Project Manager assumes responsibility for numerous operations that are manually performed with stand-alone design tools.

Project Manager performs the following functions:

- Automatically load all design resources when a project is open.
- Checks if all projects resources are available and up-to-date.
- Shows the design process flow.
- Provides button for launching applications involved in the design process.
- Provides interface to external third-arty programs.
- Places all error and status messages in the message window.
- Provides automated data transfer between tools involved in processing design.
- Provides design status information.

The Project Manager window has three main areas; messages, project files and design flow. The message window is for errors, warnings and general messages concerning the status of simulations or synthesis activities in progress. The upper left portion shows the files in the current project. By selecting the flow tab in upper right portion, a graphical representation of the core activities will take place within the Foundation Express toolset, and refers to as the design flow.

The first rectangle in the design flow provides buttons for model creation. There are three ways to construct a model. The first is by using a schematic editor and constructing hardware models from library components. The second is a state machine editor that provides a graphical interface for the construction of state machine. The third approach is using a txt editor and entering HDL code. Figure 3.1 shows the Project Manager Window.

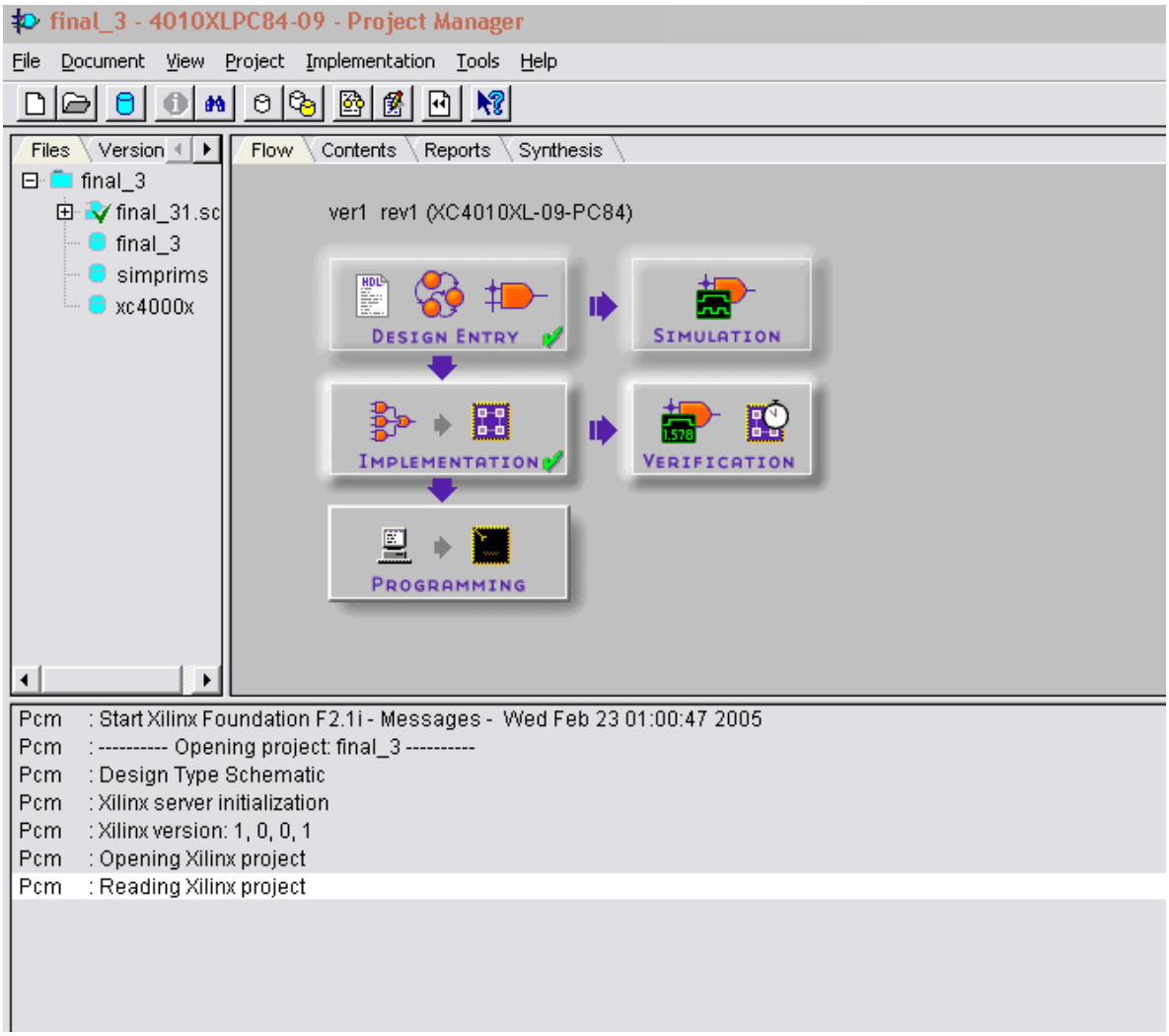

Figure 3.1: Project Manager Window

3.2 FPGA Design Flow

The design flow broadly refers to the sequence of activities encompassing various design tools that begin with some abstract specification of a design and ends with a configured FPGA. Figure 3.2 shows the FPGA design flow diagram.

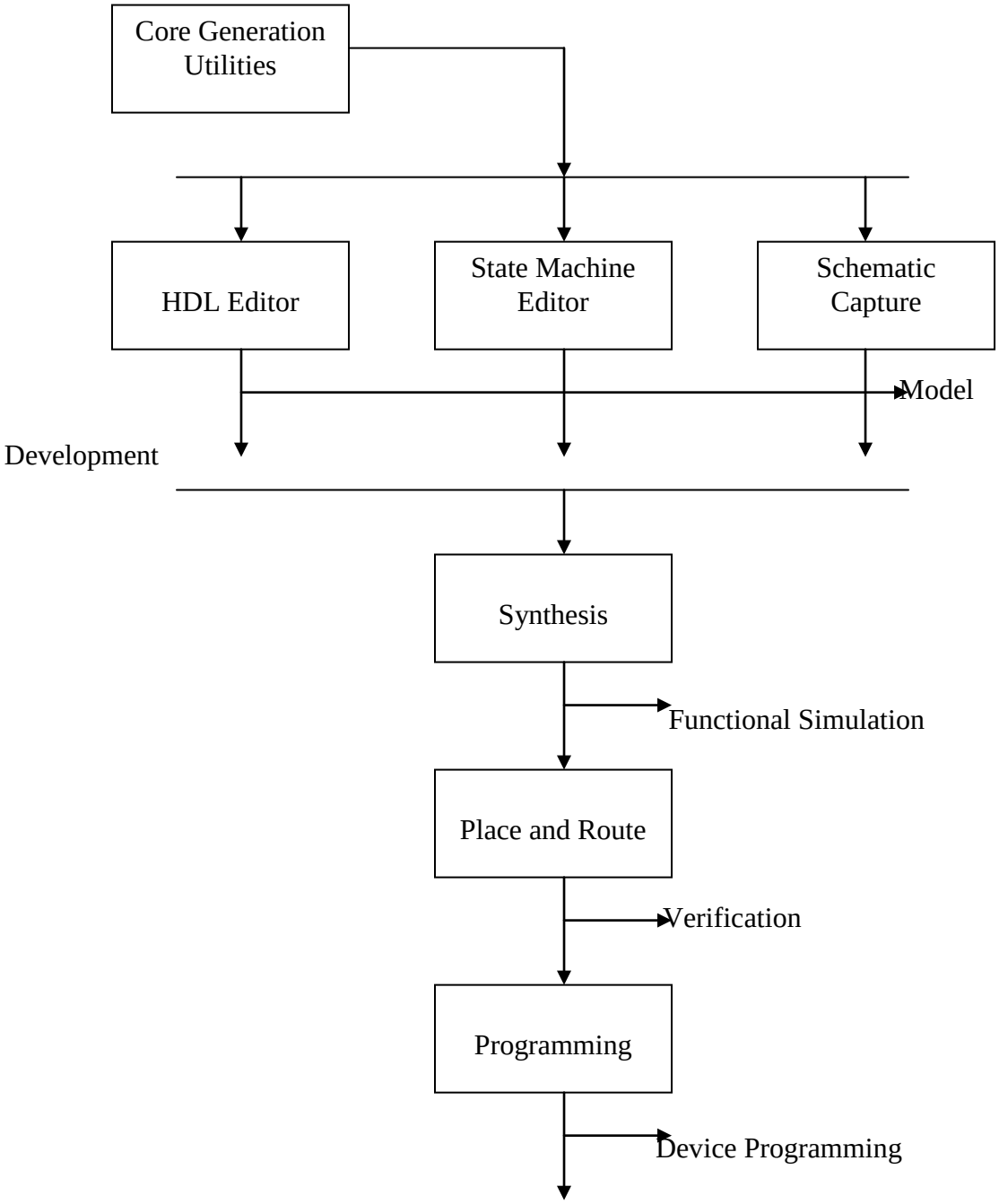

Figure 3.2: FPGA Design Flow

3.2.1 Schematic Editor

Schematic Editor is a primary design entry tool. It supports creation of multi-sheet hierarchical schematics. It keys features are:

- Support for multiple sheet, flat and hierarchical schematics.
- Integration with Logic Simulator providing for non-schematic simulation.
- Integration with non-schematics design entry tools (HDL Editor and State Diagram Editor), providing for the use of non-schematic macro.
- Import of Viewlogic schematics.
- Schematic netlist exported to XNF, EDIF and VHDL formats.
- Support for both board-level and FPGA schematics.

Schematic Editor should always be started from Project Manager. These two applications are tightly integrated, so Schematic Editor demands that Project Manager is running in the background. Schematic Editor can be started from the Project Manager in one of the following ways:

1. By clicking the **Schematic Editor** Button in the **Design Entry** area of the **Flow** tab.
2. By choosing the **Schematic Editor** from the **Design Entry** submenu of the **Tools** menu.
3. By double-clicking in the hierarchy tree the name of the schematic document that need to be opened.

If a project of a subtype Schematic is been created, Schematic Editor will automatically create a new blank sheet (schematic file) and attach it to the project as a top-level document. The file will have the name of the project ended with ‘1’. As a result, a new icon will appear in the hierarchy tree in Project Manager.

If Schematic Editor with a project that already includes some schematic files is started, Schematic Editor will automatically open all files that were in use during the previous session.

Schematic Editor is a multiple-document, means work can be done with several open schematic windows. This is particularly important for large schematics which usually consist of numerous schematic files. Figure 3.3 shows the Schematic Editor Window.

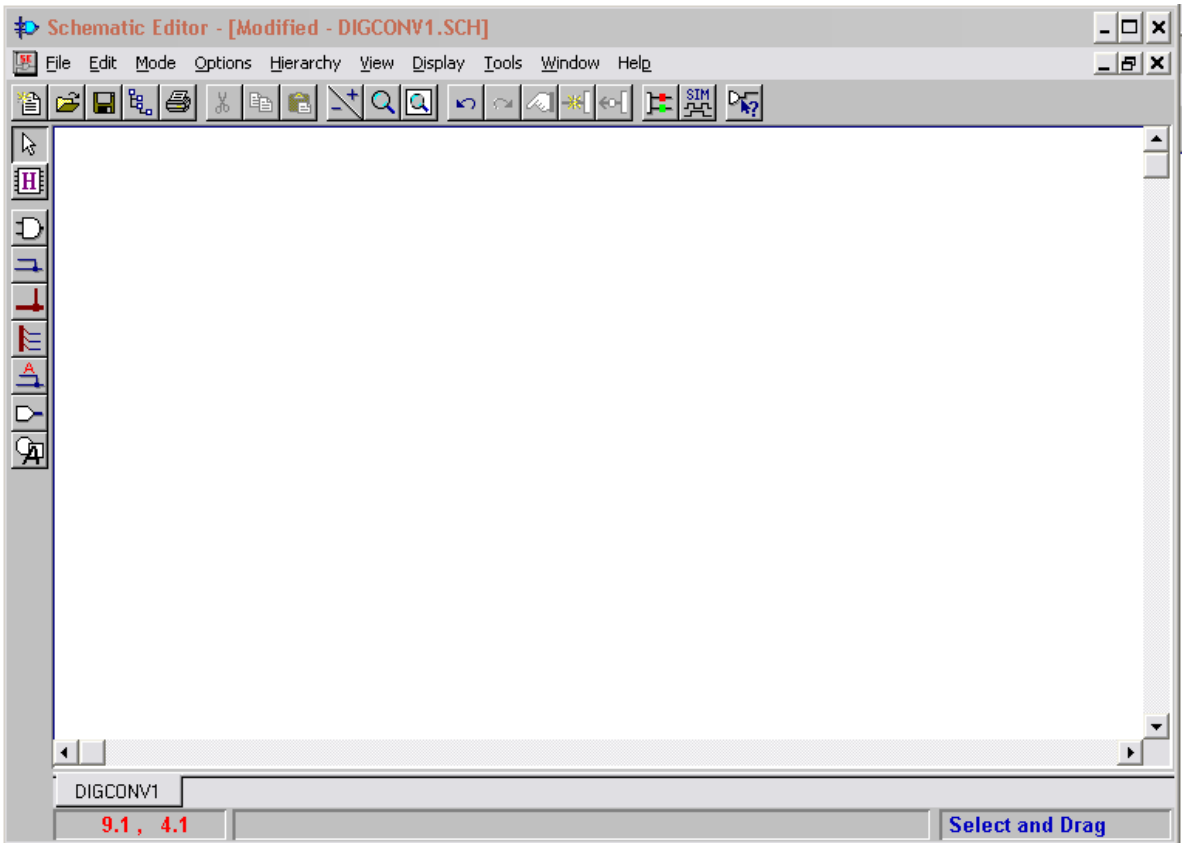

Figure 3.3: Schematic Editor Window

The New Symbol Wizard is a tool to create a macro symbol. The wizard displays consecutively five dialogue windows, in which are prompted to enter data needed to build the macro symbol. The **Next** button moves to the next symbol wizard. The **Back** button allows the user to move back to the previous one and cancel all settings in the current dialogue. Each wizard dialogue displays detailed instructions for the user. The wizard can be quit out at anytime by clicking the **Cancel** button. Figure 3.4 shows the design wizard window (for each dialogue window).

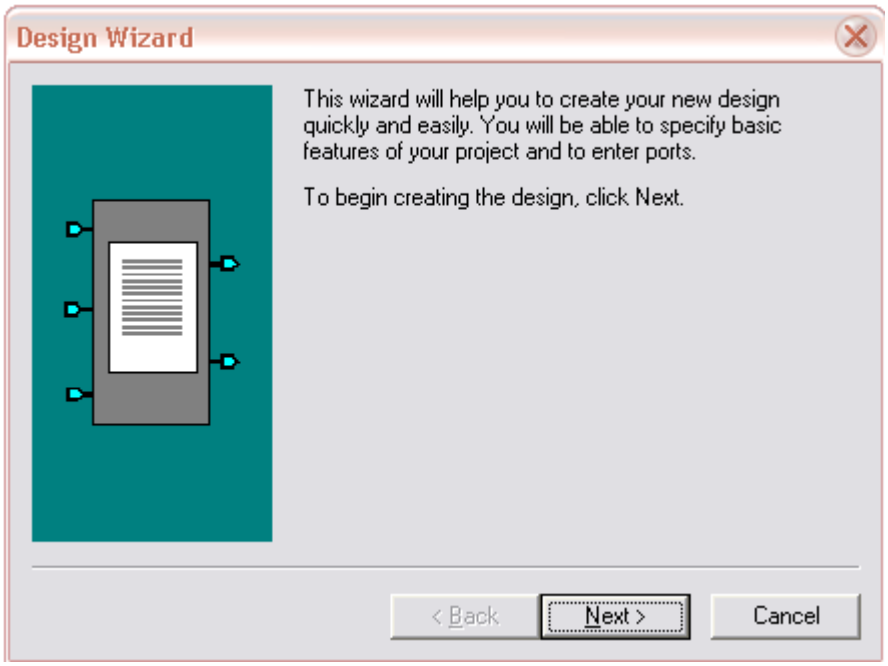

Figure 3.4a: Welcome window

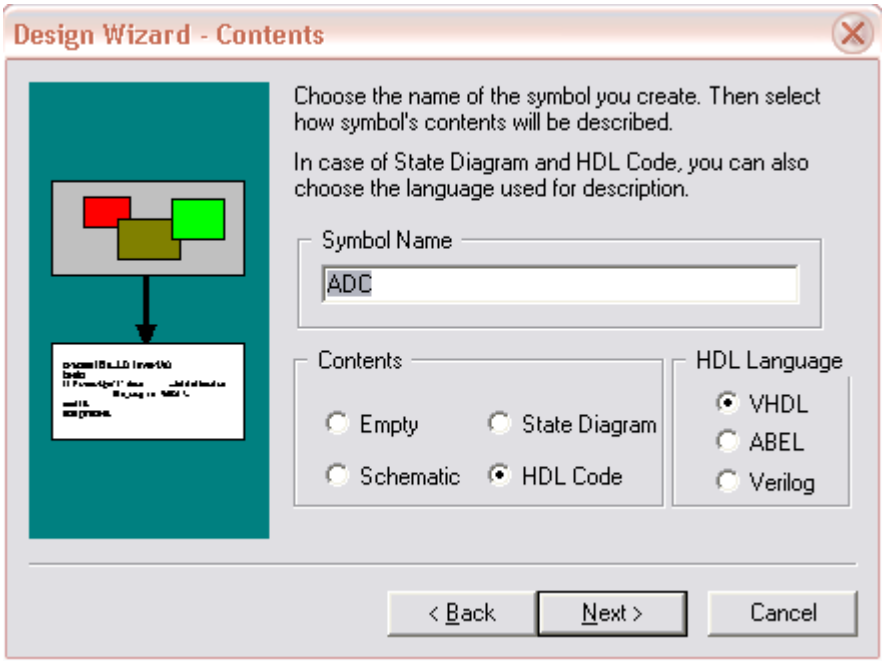

Figure 3.4b: Contents Window

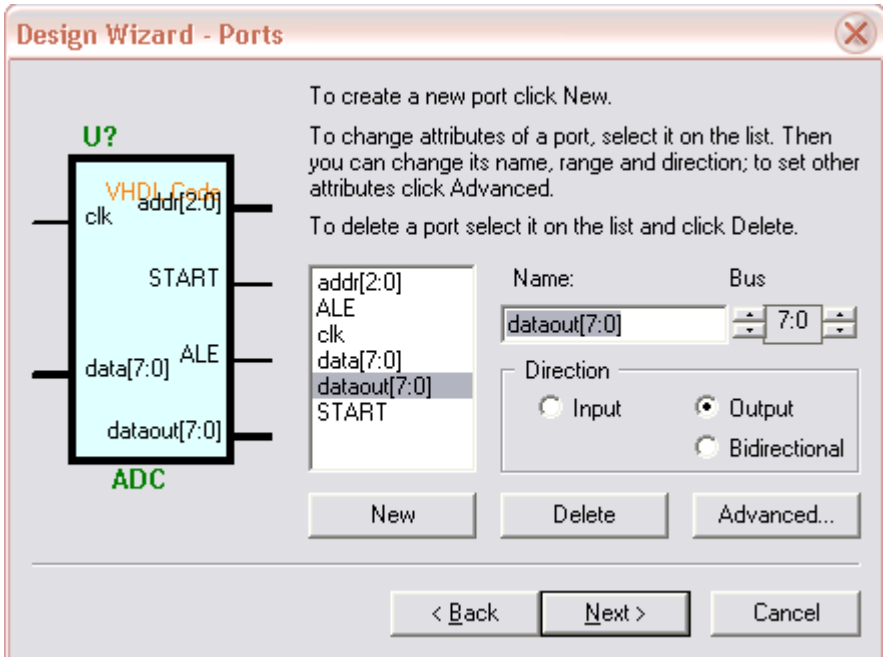

Figure 3.4c: Ports Window

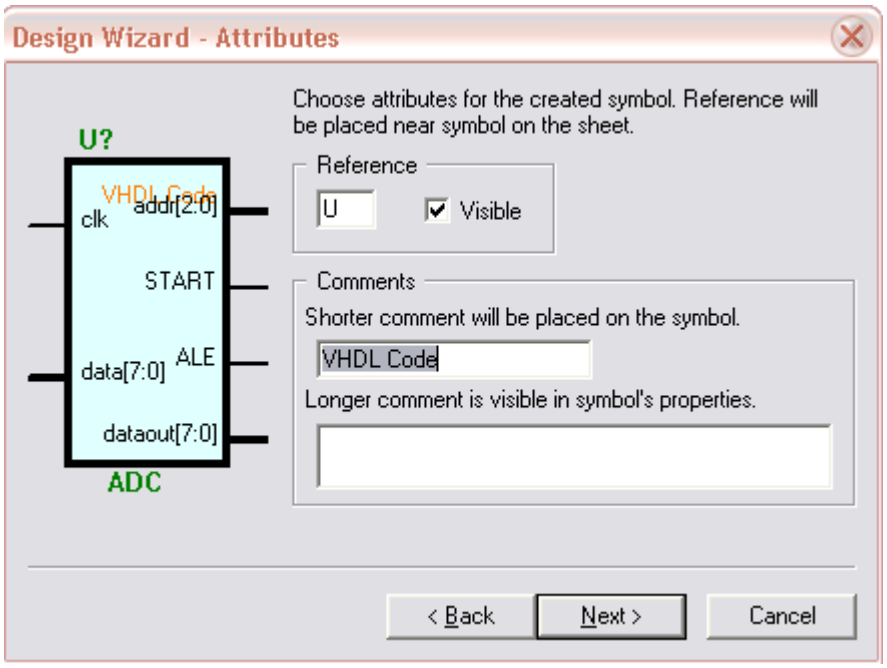

Figure 3.4d: Attributes Window

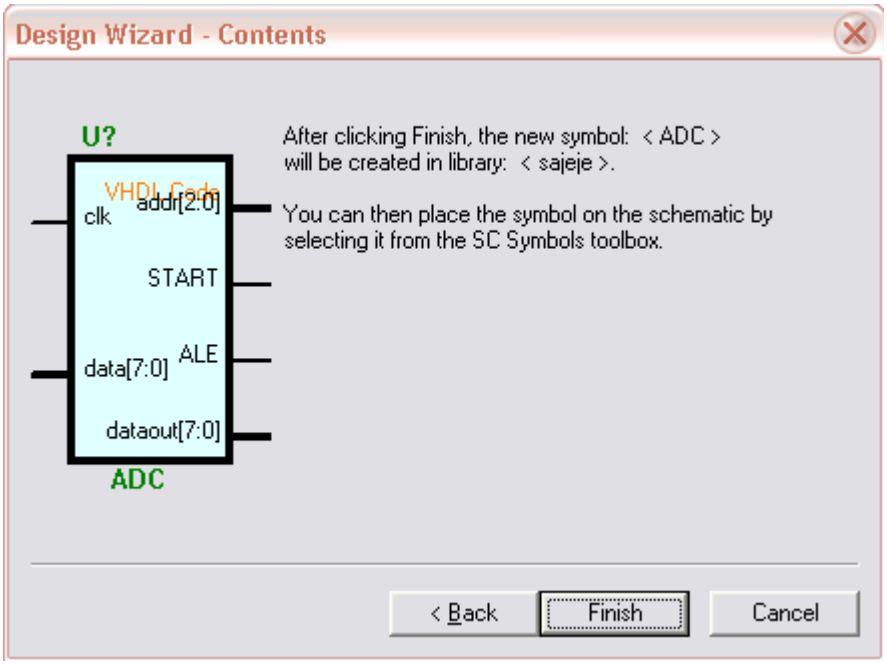

Figure 3.4e: Final Window

3.2.2 HDL Editor

The HDL Editor is a text editor designed for editing HDL source files. In addition to regular editing features, the editor provides interface to external synthesis tools, commands for hierarchy operations and syntax coloring. The syntax coloring feature supports three languages, VHDL, ABEL and Verilog.

It can be started be in two ways:

1. By choosing **Design Entry>HDL Editor** from the **Tools** menu.
2. By clicking the **HDL Entry** button in the project flowchart.

In all the cases, HDL Editor works in a non-hierarchical mode. In this mode, library data associated with the source files can not be updated. Nevertheless, saving changes in the source is possible.

In addition HDL Editor can be automatically invoked during operations related to the project hierarchy. In such situations it works in the hierarchical mode. In this mode, HDL editor can automatically update library data associated with the source files. This allows updating the macro ALB netlist and symbol after any modifications have been made to the source file. Figure 3.5 shows the HDL Editor Window.

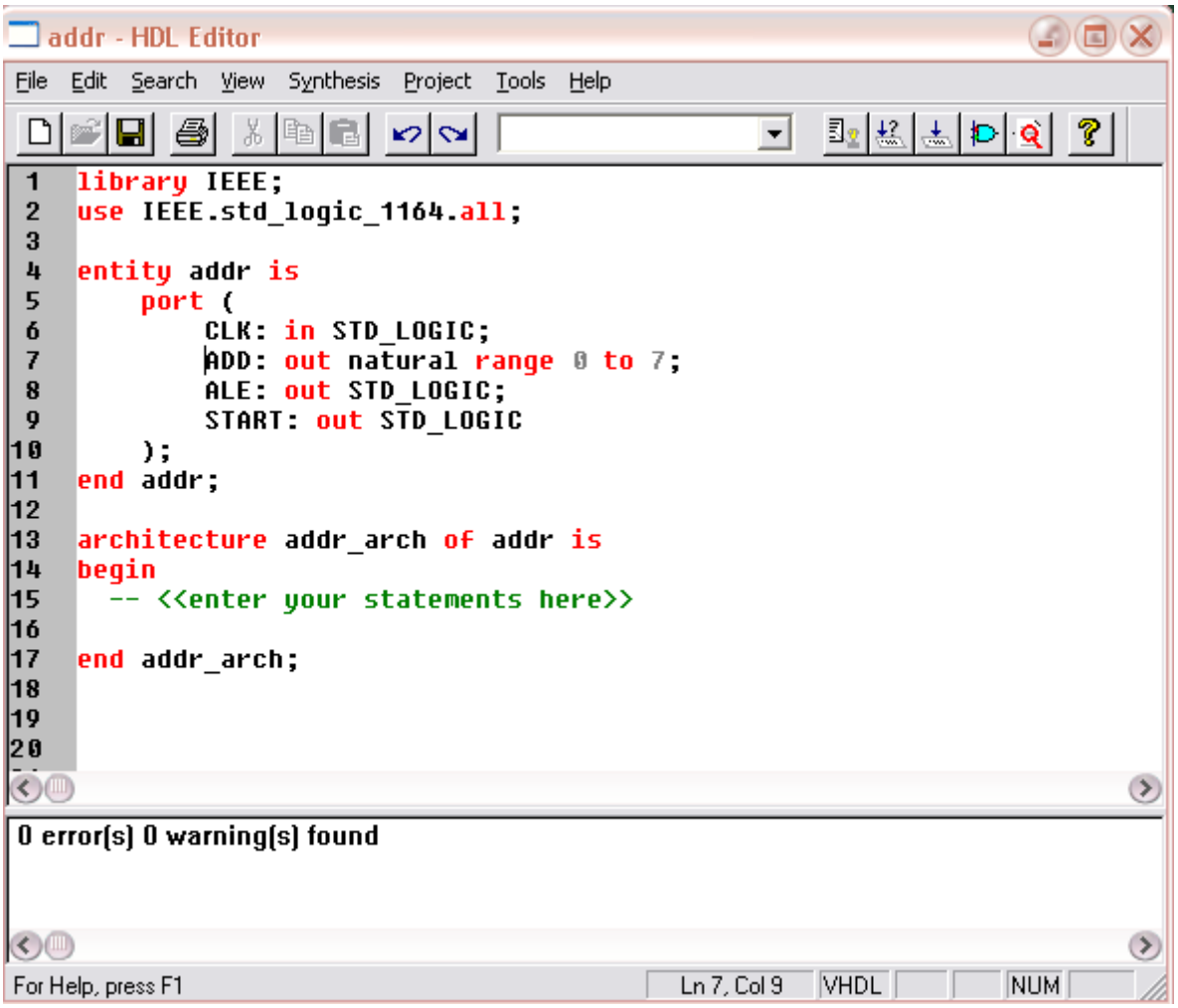

Figure 3.5: HDL Editor Window

3.2.3 Logic Simulator

The foundation Logic Simulator is a real-time interactive design analysis tool. It can work both with Foundation Schematic Editor and as a standalone tool. It key features are:

- Supports full integration with Foundation Schematic Editor.
- Accept both flat and hierarchical Xilinx XNF and EDIF netlists.
- Supports Functional and Timing simulation mode.
- Supports graphical waveform editing.
- Supports easy-to-use predefined stimulator type.
- Supports simulation of memory devices.
- Detects design timing errors such as timing violations and bus conflicts.

There are three ways of starting the Foundation Simulator from Project Manager: