CHAPTER 3

EVLVABLE HARDWARE IN FPGA

In chapter 2, the details of the evolvable hardware circuit and the Genetic unit are discussed. The functionality and design descriptions of the different modules are discussed in detail. This chapter deals with the hardware implementation issues of the EHW unit.

3.1 REASON FOR USING VHDL FOR DESCRIPTION

The functionality of each module and the entire EHW design is realized using a Hardware Descriptive Language – HDL. VHDL is used for this purpose. The functionality of each of the modules is coded using VHDL as a behavioral description. In some cases mixed coding is also followed. The selection of VHDL for hardware description for this design is quite obvious.

i) VHDL is used for coding the functionality of each of the sub modules of the VRC because of its ease in describing the behavior of any circuit.

ii) The language is more flexible in terms of describing hardware.

iii) The hierarchical modeling such as Top – down and bottom – up design methodologies are available for digital design.

iv) There are four levels of abstraction such as Behavioral, Structural (data flow level), gate level, and switch level to represent the same module offers flexibility of using all these abstractions in VHDL.
A hierarchical design approach is shown in Figure 3.1.

**Shaded**: Processes In Design Flow; **Unshaded**: Level Of Design

*Figure 3.1 Flow Chart for EHW CHIP design*
3.2 HARDWARE IMPLEMENTATION

The results obtained by implementing the EHW using VHDL are discussed in chapter 5. The implementation of the VRC module using the Xilinx architecture shows an improvement in the speed by a factor of 180. The architecture attained by including all the functional modules inside a single chip is the novelty of the design. The results show that the single chip architecture will work satisfactorily when implemented in hardware, without affecting the functionality of any of the modules. This chapter discusses the suitability of the hardware architectures of the Xilinx family for implementing the EHW.

3.2.1 IC Technologies

The flowchart in Figure 3.1 shows the steps involved in ASIC design. There are three methods of implementing the EHW design in hardware. They are explained as follows. The three types of IC technologies are:

- Full-custom VLSI.
- Semi-custom ASIC (gate array and standard cell).
- PLD (Programmable Logic Device).

a. Full-custom VLSI: All the layers of the device are optimized for an embedded system’s particular digital implementation. The steps involved in the design are: Placing transistors, Sizing transistors and routing wires. The benefits are excellent performance, small size and low power consumption. The major drawback is that it requires weeks to months for development.
b. **Semi-custom:** Lower layers of the device are fully or partially built. Designers are left with routing of wires and placing some blocks. This offers good performance, small size, less cost than a full-custom implementation. The major drawback is that it requires weeks to months for development.

### 3.2.2 Programmable Logic Devices (PLD)

A PLD is an integrated circuit chip that can be configured by the end user to realize different designs and they are re-programmable and re-usable. Smaller PLDs are programmed using special programming units. Others are programmed, in-system using JTAG. Figure 3.2 shows the categorization of PLDs.

![Figure 3.2 PLD categorization](image-url)
In a PLD, all layers already exist and the designers can purchase an IC and development board. The connections on the IC are either created or destroyed to implement desired functionality, that is, they are re-programmable. One such IC is Field Programmable Gate Array (FPGA). The benefits offered by the PLDs are very low cost and almost instant IC availability and rapid prototyping facility to verify the functionality of the design and therefore complements the design verification efforts. Table 3.1 gives the characteristics of PLDs.

**Table 3.1 Characteristics of Programmable Logic Devices**

<table>
<thead>
<tr>
<th>Sl. No.</th>
<th>Criteria</th>
<th>CPLD</th>
<th>FPGA</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Architecture</td>
<td>PLA based More Combinational</td>
<td>LUTs-Register based Registers + RAM</td>
</tr>
<tr>
<td>2</td>
<td>Density</td>
<td>Low-to-medium 0.5-100K logic gates</td>
<td>Medium to high 100K to 10M system gates</td>
</tr>
<tr>
<td>3</td>
<td>Performance</td>
<td>Predictable timing &gt; 200 MHz</td>
<td>Application &amp; Utilization Dependent</td>
</tr>
<tr>
<td>4</td>
<td>Interconnect</td>
<td>“Crossbar”</td>
<td>“Segmented”</td>
</tr>
<tr>
<td>5</td>
<td>Application</td>
<td>Glue Logic</td>
<td>Hardware Emulation</td>
</tr>
</tbody>
</table>

3.2.2.1. **Programmable Logic Array (PLA)**

Programmable Logic Arrays are designed using AND and OR gates. It is best suited for implementing complex combinatorial logic and eliminates
discrete gates in the design and they are re-programmable and provide flexibility. A simple programmable AND/OR array is shown in Figure 3.3.

3.2.2.2. Complex Programmable Logic Devices (CPLD)

CPLDs consist of a number of macro cells (PLA based architecture) typically in the range of 32-512 and has a capacity up to 100K gates and is an ideal candidate for low-medium density designs. CPLDs are best suited for combinatorial high-speed designs up to about 200 MHz. They offer predictable performance because of constant delay between 2 Macro cells. The vendor provides one step synthesis and mapping tool for design entry using Verilog/VHDL and the device can be programmed using JTAG. The programme is erasable and re-programmable in system. The range of device utility is typically in the range of 100 user I/Os.

Figure 3.3 Programmable Logic Array
3.2.2.3 Field Programmable Gate Array (FPGA)

FPGAs are preferred to CPLDs for implementation of complex high-density design for the following reasons:

i) FPGAs provide cost effective solution to quickly prototype a design.

ii) Validating the design on the FPGA hardware minimizes the risk when entering into fabrication.

iii) FPGA hardware emulation enables development of software much ahead of implementation.

iv) Array of logic cells called Configurable Logic Blocks (CLB) connected via routing channels can be programmed.

v) The gate densities from 40K to 10M Gates make this ideal for high-density designs.

vi) Supports huge number of I/O standards like TTL, CMOS, LVCMOS, PCI, LVDS, etc (E.g.: Xilinx Virtex-EM FPGAs support 19 different single-ended and 6 different differential standards).

vii) FPGAs with user pin outs greater than 100 user I/Os are available.

viii) FPGAs have built-in components like RAMs, Multipliers, Clock, Delay Locked Loops, ALU, Content Addressable Memories (CAM), Processors, and Intellectual Property (IP) cores are available for the user apart from logic cells.

ix) FPGAs consist of array of Configurable Logic Blocks. Each CLB comprises of 4 Slices and is tied to a Switch Matrix to
access the Global Routing Matrix. Each Slice contains 4-Input
Look Up Tables (Function Generators) and is capable of
implementing any Boolean function.

x) FPGAs contain multiplexers, Carry Logic, Arithmetic Logic
Gates and two Storage Elements (Flip-Flop/Latch).

xi) Plenty of Fast 18x18 Multipliers (in some FPGAs) are available
for DSP Applications.

xii) FPGAs have large amounts of 18K-Bit Select RAM all over the
chip (Synchronous operation, configurable as Single or Dual
ported, configurable in different sizes like 2Kx9, 16Kx1,
4Kx4, etc) are available.

xiii) FPGAs have multiple Digital Clock Managers (DCM)
available to de-skew and phase align all the internal clocks with
respect to the input clock and they offer flexible Frequency
Synthesis using DCM

xiv) Multiple Global Clock Buffers are available for Clock
distribution in the chip.

3.2.3 Advantages of using FPGAs for implementing VRC Unit of
EHW

- With FPGAs it is possible for full functional verification
  before fabricating the chip.

- Re-Programmable and hence it is possible to fix and verify
  functionality in the design stage.

- Hardware Emulation of the chip is possible with FPGA.
  Figure 3.4 shows the hardware and software verification
  involved in testing the design.
3.3 STEPS INVOLVED IN FPGA EMULATION

- FPGA identification.
- FPGA based circuit board with other debug accessories to verify the design in hardware.
- Synthesis Tool (Convert RTL design to the targeted FPGA gates).
- Place & Route Tool (mostly from the FPGA vendor) that will place and route the synthesized gate net list.
- FPGA programming tools (e.g.: JTAG/parallel port cable).

Figure 3.4 Hardware & Software Verification
3.3.1 Hardware Selection Issues

Prior to implementation as a chip, the single chip VRC is to be tested in a programmable logic device in order to analyze the real time hardware functionalities of the new design. There are a number of programmable logic devices available. Hence the selection of a PLD becomes an important issue for hardware realization.

Basically, the selection of a PLD for a specific application is based on one or more of the following requirements.

i) Number of memory elements used in the design.
ii) Approximate Gate counts for the Design.
iii) Complexity of the design.
iv) Availability of the associated chips or boards for testing.
v) Nature of inputs and outputs.

The EHW chip that is to be designed has got 8 memory elements each of 32-bit size. According to the design specifications, all data that is written or read from any memory of the chip is 32-bit information per clock cycle (1ns). The design requires at least 8 RAMs of size of 32 bit each.

Other hardware components instantiated in the design are:

- Counters are instantiated in all the fifteen modules of the design.
- Registers are instantiated in all the modules of the design.
- There are a number of flags used in the design.
- There are a number of 32 bit busses used in the design.

The complexity of the design can be seen from the number of hardware elements that are instantiated. Based on the complexity of the
design, an approximate prediction of gate counts for implementing the design in a programmable matrix would be at least 1 to 1.5 lakhs.

The discussion stated above, does not permit use of PLA or CPLD for testing this complex design. An FPGA with high gate counts and with enormous other associated features, would be much suitable for implementation and testing of the EHW chip. Accordingly, the FPGA Xilinx Virtex series (XCVfg680) that has enough features for implementing the design, is selected. The following section describes the various required features, architectural and functional descriptions of the device.

3.4 DESCRIPTION OF VIRTEX-EM FPGA ARRAY
3.4.1 VIRTEX-EM FPGA (XCVfg680)

The Virtex™-E Field-Programmable Gate Array gives high performance, abundant logic resources, and a rich feature set. The chip offers densities ranging from 15,000 to 100,000 system gates, as shown in Table 3.2. System performance is supported up to 200 MHz. VIRTEX-EM FPGA devices deliver more gates, I/Os, and features. Features include 10 block RAM (40K bits), distributed RAM (to 38,400 bits), 2,700 Logic cells, 600 CLBs, 16 selectable I/O standards, 1,00,000 system gates (including logic and RAM) and four DLLs.

The device is fully supported by powerful Xilinx development system

- Foundation ISE Series: Fully integrated software
- Alliance Series: For use with third-party tools
- Fully automatic mapping, placement, and routing
3.4.2 Functional Description of VIRTEX-EM FPGA (XCVfg680)

The architecture of Virtex™-E FPGA is shown in Figure 3.5. The FPGA has a regular, flexible, programmable architecture of Configurable Logic Blocks (CLBs), surrounded by a perimeter of programmable Input/Output Blocks (IOBs). There are four Delay-Locked Loops (DLLs), one at each corner of the die. Two columns of block RAM lie on opposite sides of the die, between the CLBs and the IOB columns. A powerful hierarchy of versatile routing channels interconnects these functional elements. Loading configuration customizes Virtex FPGA data into internal static memory cells. The device offers unlimited reprogramming cycles. The stored values in these cells determine logic functions and interconnections implemented in the FPGA. The configuration data can be read from an external serial PROM using master serial mode, or written into the FPGA in slave serial, slave parallel, or boundaries scan modes. The device provides system clock rates up to 200 MHz. In addition to the conventional benefits of high-volume programmable logic solutions, Virtex FPGA also offer on-chip synchronous single-port and dual-port RAM (block and distributed form), DLL clock drivers, programmable set and reset on all flip-flops, and fast carry logic.

Table 3.2 Device Specifications for Virtex-EM family

<table>
<thead>
<tr>
<th>Device</th>
<th>Logic Cells</th>
<th>System Gates (Logic and RAM)</th>
<th>CLB Array (RxC)</th>
<th>Total CLBs</th>
<th>Maximum Available User I/O(1)</th>
<th>Total Distributed RAM Bits</th>
<th>Total Block RAM Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCV15</td>
<td>432</td>
<td>15,000</td>
<td>8 x 12</td>
<td>96</td>
<td>66</td>
<td>6,144</td>
<td>16K</td>
</tr>
<tr>
<td>XCV30</td>
<td>972</td>
<td>30,000</td>
<td>12 x 18</td>
<td>216</td>
<td>132</td>
<td>13,824</td>
<td>24K</td>
</tr>
<tr>
<td>XCV50</td>
<td>1,728</td>
<td>50,000</td>
<td>16 x 24</td>
<td>384</td>
<td>176</td>
<td>24,576</td>
<td>32K</td>
</tr>
<tr>
<td>XCV100</td>
<td>2,700</td>
<td>100,000</td>
<td>20 x 30</td>
<td>600</td>
<td>196</td>
<td>38,400</td>
<td>40K</td>
</tr>
<tr>
<td>XCV150</td>
<td>3,888</td>
<td>150,000</td>
<td>24 x 36</td>
<td>864</td>
<td>260</td>
<td>55,296</td>
<td>48K</td>
</tr>
<tr>
<td>XCV200</td>
<td>5,292</td>
<td>200,000</td>
<td>28 x 42</td>
<td>1,176</td>
<td>284</td>
<td>75,264</td>
<td>56K</td>
</tr>
</tbody>
</table>
3.4.3 Architectural Description of Virtex-EM Array

The VIRTEX-EM user-programmable gate array, shown in Figure 3.5 is composed of five major configurable elements:

- IOBs provide the interface between the package pins and the internal logic.
- CLBs provide the functional elements for constructing most logic
- Dedicated block RAM memories of 4096 bits each.

![Figure 3.5 Basic Virtex-EM FPGA block Diagram](image-url)
Clock DLLs for clock-distribution delay compensation and clock domain control.

Multi-level interconnect structure.

As can be seen in Figure 3.5, the CLBs form the central logic structure with easy access to all support and routing structures. The IOBs are located around all the logic and memory elements for easy and quick routing of signals on and off the chip. Values stored in static memory cells control all the configurable logic elements and interconnect resources. These values load into the memory cells on power-up, and can reload if necessary to change the function of the device. Each of these elements will be discussed in detail in the following sections.

3.4.3.1 Input/Output Block

The Virtex-EM IOB, as seen in Figure 3.6, features inputs and outputs that support a wide variety of I/O signaling standards. These high-speed inputs and outputs are capable of supporting various memory and bus interfaces. The three IOB registers function either as edge-triggered D-type flip-flops or as level-sensitive latches. Each IOB has a clock signal (CLK) shared by the three registers and independent Clock Enable (CE) signals for each register.

3.4.3.2 Configurable Logic Block (CLB)

The basic building block of the Virtex-EM CLB is the logic cell (LC), and is shown in Figure 3.7. An LC includes a 4-input function generator, carry logic, and storage element. Output from the function generator in each LC drives the CLB output and the D input of the flip-flop. Each CLB contains four LCs, organized in two similar slices; a single slice is
shown in Figure 3.7. In addition to the four basic LCs, the Virtex-EM CLB contains logic that combines function generators to provide functions of five or six inputs.

### 3.4.3.3 Look-Up Tables

Virtex-EM function generators are implemented as 4-input look-up tables (LUTs). In addition to operating as a function generator, each LUT can provide a 16 x 1-bit synchronous RAM. Furthermore, the two LUTs within a slice can be combined to create a 16 x 2-bit or 32 x 1-bit synchronous RAM, or a 16 x 1-bit dual-port synchronous RAM. The LUT can also provide a 16-bit shift register that is ideal for capturing high-speed or burst-mode data. This mode can also be used to store data in applications such as Digital Signal Processing.

![Figure 3.6 Input Output Block of Virtex-EM FPGA](image-url)
3.4.3.4 Storage Elements

Storage elements in the Virtex-EM slice can be configured either as edge-triggered D-type flip-flops or as level-sensitive latches. The D inputs can be driven either by function generators within the slice or directly from slice inputs, bypassing the function generators.

Figure 3.7 Configurable Logic Blocks (CLB)
3.4.3.5 Block RAM

Virtex-EM FPGA incorporates several large block RAM memories. This complements the distributed RAM Look-Up Tables (LUTs) that provide shallow memory structures implemented in CLBs. Block RAM memory blocks are organized in columns. The device contains two such columns, one along each vertical edge. These columns extend the full height of the chip. Each memory block is four CLBs high, and consequently, a Virtex-EM device with eight CLBs high will contain two memory blocks per column, and a total of four blocks. Each block RAM cell, as illustrated in Figure 3.8, is a fully synchronous dual-ported 4096-bit RAM with independent control signals for each port. The data widths of the two ports can be configured independently, providing built-in bus-width conversion. The block RAM also includes dedicated routing to provide an efficient interface with both CLBs and other block RAMs. Table 3.3 illustrates the block RAM present in Virtex-EM FPGA and table 3.4 gives the port aspect ratios of the block RAM.

<table>
<thead>
<tr>
<th>Virtex-EM Device</th>
<th>% of Blocks</th>
<th>Total Block RAM Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>XCV15</td>
<td>4</td>
<td>16K</td>
</tr>
<tr>
<td>XCV30</td>
<td>6</td>
<td>24K</td>
</tr>
<tr>
<td>XCV50</td>
<td>8</td>
<td>32K</td>
</tr>
<tr>
<td>XCV100</td>
<td>10</td>
<td>40K</td>
</tr>
<tr>
<td>XCV150</td>
<td>12</td>
<td>48K</td>
</tr>
<tr>
<td>XCV200</td>
<td>14</td>
<td>56K</td>
</tr>
</tbody>
</table>
Figure 3.8 Block diagram of dual port Block RAM

Table 3.4 Block RAM port aspect ratios

<table>
<thead>
<tr>
<th>Width</th>
<th>Depth</th>
<th>ADDR Bus</th>
<th>Data Bus</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>4096</td>
<td>ADDR&lt;11:0&gt;</td>
<td>DATA&lt;0&gt;</td>
</tr>
<tr>
<td>2</td>
<td>2048</td>
<td>ADDR&lt;10:0&gt;</td>
<td>DATA&lt;1:0&gt;</td>
</tr>
<tr>
<td>4</td>
<td>1024</td>
<td>ADDR&lt;9:0&gt;</td>
<td>DATA&lt;3:0&gt;</td>
</tr>
<tr>
<td>8</td>
<td>512</td>
<td>ADDR&lt;8:0&gt;</td>
<td>DATA&lt;7:0&gt;</td>
</tr>
<tr>
<td>16</td>
<td>256</td>
<td>ADDR&lt;7:0&gt;</td>
<td>DATA&lt;15:0&gt;</td>
</tr>
</tbody>
</table>

3.4.3.6 Programmable Routing Matrix

It is the longest delay path that limits the speed of any worst-case design. Consequently, the Virtex-EM FPGA routing architecture and its place-and-route software were defined in a single optimization process. This joint optimization minimizes long-path delays, and yields the best system performance. The joint optimization also reduces design compilation times.
because the architecture is software-friendly. Design cycles are correspondingly reduced due to shorter design iteration times.

3.4.3.7 Local Routing

The local routing resources, as shown in Figure 3.9, provide the following three types of connections:

- Interconnections among the LUTs, flip-flops, and General Routing Matrix (GRM)
- Internal CLB feedback paths that provide high-speed connections to LUTs within the same CLB, chaining them together with minimal routing delay
- Direct paths that provide high-speed connections between horizontally adjacent CLBs, eliminating the delay of the GRM

![Figure 3.9 Local Routing in Virtex-EM](image)
3.4.3.8 Clock Distribution

The FPGA provides high-speed, low-skew clock distribution through the primary global routing resources described above. Four global buffers are provided, two at the top center of the device and two at the bottom center. These drive the four primary global nets that in turn drive any clock pin. Four dedicated clock pads are provided, one adjacent to each of the global buffers. The input to the global buffer is selected either from these pads or from signals in the general purpose routing.

3.4.3.9 Delay-Locked Loop (DLL)

Associated with each global clock input buffer is a fully digital Delay-Locked Loop (DLL) that can eliminate skew between the clock input pad and internal clock-input pins throughout the device. Each DLL can drive two global clock networks. The DLL monitors the input clock and the distributed clock, and automatically adjusts a clock delay element. Additional delay is introduced such that clock edges reach internal flip-flops exactly one clock period after they arrive at the input. This closed-loop system effectively eliminates clock-distribution delay by ensuring that clock edges arrive at internal flip-flops in synchronism with clock edges arriving at the input. In addition to eliminating clock-distribution delay, the DLL provides advanced control of multiple clock domains. The DLL provides four quadrature phases of the source clock which can be used to double the clock, or divide the clock by 1.5, 2, 2.5, 3, 4, 5, 8, or 16. It has six outputs. In order to guarantee that the system clock is operating correctly prior to the FPGA starting up after configuration, the DLL can delay the completion of the configuration process until after it has achieved lock.
3.5 FPGA BASED CIRCUIT BOARD WITH DEBUG
ACCESSORIES

3.5.1 Selection of the Prototyping board

After selecting a suitable FPGA for testing the design, the next step is to select an appropriate prototyping board. This is more important due to the fact that the testing can be done only if appropriate prototyping board has all associated chips. The selection is usually done based on the following criteria.

i. Nature of inputs and Outputs.
ii. Complexity of the design.
iii. Testing method used.
iv. Software for downloading the design.
v. Interfacing units.

i) **Nature of Inputs and Outputs**: The nature of inputs to the VRC processor is stream of data bits (for example 9 bytes) and the output will be eight bits replacing the center pixel in the 3x3 kernel window. Since the design requires stream of data to be forced, on - off switches cannot be used for giving inputs. Similarly, the outputs are also stream of bits, which cannot be viewed using any output indicator such as LEDs. Even when Logic analyzer is used there must be enough number of channels for verification of the outputs. Therefore, for testing the design, it is essential that the input stream of bits be stored in an external RAM and the same is forced into the design, and the output be again stored in the same RAM in a different location.
ii) Complex designs require more number of gates in FPGA and other accessories.

iii) **Testing Method:** A test control block is designed for this purpose.

iv) **Software for downloading:** The downloading of the configuration bits is done using software.

v) **Interfacing units:** The interfacing units are necessary for communication of data between the computer and the FPGA.

### 3.5.2 Layout of the FPGA Prototyping Board

The Figure 3.10 shows the layout of the prototyping board used for fusing the design into FPGA. The board has the following modules.

i) 16 MB SDRAM (8M x 16)

ii) Xilinx XCVfg680 Virtex-EM FPGA

iii) 256KB flash memory

iv) CPLD Xilinx XC9572

v) Parallel port interface

vi) 100 MHz Programmable Oscillator

vii) Seven segment display and four DIP switches

viii) PC loaded with bit stream loading software

ix) Push Button Switches
This section describes the important accessories of the prototyping board and shows how the I/O of the FPGA and CPLD are connected to the rest of the circuitry.

i. A 100-K gate Xilinx VIRTEX FPGA (XCVfg680) in a 144-pin QFP package. The FPGA is the main repository of programmable logic on the board.

ii. A Xilinx XC9572XL CPLD that is used to manage the configuration of the FPGA via the parallel port. The CPLD also controls the programming of the Flash RAM.
3.6 16 MB SDRAM

A Hynix HY57V281620AT-H SDRAM with 16 Mbytes of storage (8M × 16) is connected to the FPGA as shown in Figure 3.11. The clock signal to the SDRAM is also re-routed back to a dedicated clock input of the FPGA. This makes it easy to synchronize the internal operations of the FPGA with the SDRAM operations.

![Figure 3.11 FPGA and SDRAM connection diagram](image)

3.7 256 Kbytes Flash RAM

An Atmel AT49F002 Flash RAM with 256 Kbytes of storage (256K × 8) is connected to both the FPGA and CPLD as shown in Figure 3.12. The CPLD and FPGA both have access to the Flash RAM. Typically, the
CPLD will program the Flash with data passed through the parallel port. If the data is an FPGA configuration bit stream, then the CPLD can be configured to program the FPGA with the bit stream from Flash whenever the board is powered up. After power-up, the FPGA can read and/or write the flash. (Of course, the CPLD and FPGA have to be programmed such that they do not conflict if both are trying to access the Flash.) The Flash can be disabled by raising the CE pin to logic 1 in which case the I/O lines connected to the Flash can be used for general-purpose communication between the FPGA and the CPLD.

Figure 3.12 Flash RAM connection with CPLD and FPGA
3.8 VGA MONITOR INTERFACE

The FPGA can generate a video signal for display on a Video Graphics Adaptor (VGA) monitor interface, which is shown in Figure 3.13. When the FPGA is generating VGA signals, it outputs two bits of red, green, and blue color information to a simple resistor-ladder DAC. The outputs of the DAC are sent to the RGB inputs of a VGA monitor along with the horizontal and vertical sync pulses (/HSYNC, /VSYNC) from the FPGA.

![Figure 3.13 VGA Interface with FPGA](image)

3.9 PS/2 PORT

The board provides a PS/2-style interface (mini-DIN connector J4) to either a keyboard or a mouse. The FPGA receives two signals from the PS/2 interface: a clock signal and a serial data stream that is synchronized with the falling edges on the clock.

3.9.1 Parallel Port Interface

The parallel port is the main interface for communicating with the board is shown in Figure 3.14. Control line C0 goes directly to the DS1075 oscillator and is used for setting the divisor as described previously, and status
line S6 connects directly to the FPGA for use as a communication line from the FPGA back to the PC. The CPLD handles the fifteen remaining active lines of the interface to the parallel port. Eleven of the active lines of the parallel port connect to General-purpose I/O pins on the CPLD. Three of the parallel port control lines, C1–C3, connect to the JTAG pins through which the CPLD is programmed. The C1 control line clocks configuration data presented on the C3 line into the CPLD while the C2 signal steers the actions of the CPLD programming state machine.

Figure 3.14 Parallel Port Interface

Meanwhile, information from the CPLD returns to the PC through status line S7. The eight data lines, D0–D7, and the remaining three status lines, S3–S5, connect to general-purpose pins of the CPLD. The CPLD can be
programmed to act as an interface between the FPGA and the parallel port. Schmitt-trigger inverters are inserted into the D1 line so it can carry a clean clock edge for use by any state machine programmed into the FPGA/ CPLD. The CPLD connects to the configuration pins of the Virtex-EM FPGA so that it can pass configuration bit streams from the parallel port to the FPGA. The actual configuration data is presented on to the FPGA on the same 8-bit bus that connects the CPLD, Flash, seven-segment LED and FPGA. The CPLD also drives the configuration pins (CCLK, /PROGRAM, /CS, and /WR) of the FPGA that control the loading of a bit stream. The CPLD uses the M0 input of the FPGA to select either the slave-serial or master-select configuration mode (M1 and M2 are already hard-wired to VCC and GND, respectively).

The CPLD can monitor the status of the bit stream download through the /INIT, DONE, and BSY/DOUT pins of the FPGA. The CPLD also has access to the FPGA JTAG pins: TCK, TMS, TDI, TDO. The TMS, TDI, and TDO pins share the connections with the BSY/DOUT, /CS, and /WR pins. With these connections, the CPLD can be programmed with an interface that allows configuration of the Virtex-EM FPGA through the Xilinx JTAG Programmer software utility.

The FPGA sends data back to the PC by driving logic levels onto pins 40, 29 and 28 which passes through the CPLD and onto the parallel port status lines S3, S4 and S5, respectively. Conversely, the PC sends data to the FPGA on parallel port data lines D0–D7 and the data passes through the CPLD and ends up on FPGA pins 50, 48, 42, 47, 65, 51, 58 and 43, respectively. The CPLD also drives the decimal point of the LED display to indicate when the FPGA is configured with a valid bit stream.
3.10 SYNTHESES AND GENERATION OF BIT FILE

3.10.1 Design Flow

Xilinx Synthesis System – Overview

The Xilinx Synthesis system is a software tool, which has the design flow modules that are shown in Table 3.5. The flow chart in Figure 3.16 shows the detailed procedure involved in synthesizing and generating the bit stream for the design using Xilinx Synthesis technology (XST). The following section describes the method adopted for downloading the design into the target device.

Table 3.5 Xilinx Synthesis System Tools

<table>
<thead>
<tr>
<th>Design entry</th>
<th>Synthesis</th>
<th>Simulation</th>
<th>Implementation</th>
<th>Programming</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Project Navigator</td>
<td>Xilinx synthesis Technology (XST)</td>
<td>1. HDL Bencher</td>
<td>1. Chip Viewer</td>
<td>1. IMPACT (Intelligent Multipurpose Programmable and Configuration Tool)</td>
</tr>
<tr>
<td>2. HDL Editor</td>
<td></td>
<td>2. Modelsim Simulator</td>
<td>2. X – Power</td>
<td></td>
</tr>
<tr>
<td>4. Schematic editor</td>
<td></td>
<td></td>
<td></td>
<td>2. PROM file formatter</td>
</tr>
<tr>
<td>5. Constraints Editor</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6. Floor plan editor</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The design entry tools of the XST are used for entering the design modules in the form of Verilog HDL codes.
3.10.2 RTL description

The behavioral description of the functional modules is converted into Register Transfer Level (RTL) coding. The VRC is modeled using VHDL using synthesizable constructs, specific for Xilinx Synthesis system. For example, blocking assignments are used for combinational logics and for variables that are assigned and used, all within an always statement. Non-blocking assignments are used for modeling sequential logics. The design consists of more sequential logics when compared to combinational logic.

3.10.3 Synthesis

Synthesis is a process of creating gate level net list from a register transfer level model of a circuit described in Verilog HDL. That is, synthesis involves conversion of RTL Design (Verilog/VHDL) into gates present in the target FPGA library. This is shown in Figure 3.15. This is done using a synthesis tool, which is software that is provided by the chip vendor. The Xilinx synthesis system used in this design as an intermediate step generates a net list that is comprised of register transfer level blocks such as flip-flops, registers, adders, and subtractors, arithmetic logic units and multiplexers, interconnected by wires. In such cases, a module builder is used to build or acquire from a library of predefined components, each of the required RTL blocks in the user specified target technology (here, Xilinx FPGA).

![Figure 3.15 Synthesis Extractions](image-url)
Figure 3.16 Flow Chart showing steps in Synthesis of VRC (contd..)
The synthesis system has a mapping mechanism or a construction mechanism that translates the VHDL elements into their corresponding hardware elements using the library of the target device.