Implementation of NRZI Encoder And Decoder Using Spartan 3A

Jasmine Kujur
Dept of Electronics and Communication Engineering
Delhi Technological University, ShahbadDaulatpur,Delhi-42,India

Abstract- *Non return to zero inverted is one of the line coding techniques. NRZI is a method of mapping a binary signal to a physical signal for transmission over some transmission media. The two level NRZI signal has a transition at a clock boundary if the bit being transmitted is a logical 1, and does not have a transition if the bit being transmitted is a logical 0. In NRZI, a method for transmitting and recording data so that it keeps the sending and receiving clock synchronized. This is especially helpful in situations where bit stuffing is employed, the practice of adding bits to a data stream so it confirms with communication protocol. This added bits can create a long string of similar bits, which is registered to the receiver as a single, unchanging voltage since clock adjust on voltage changes the large behind true time. It ensures that after a zero bit appears, the voltage will immediately switch to one bit voltage level. The voltage level changes allow sending and receiving clock to synchronize. It is used for serial communication between a devices. NRZI uses the presence and absence of a transition to signify bit. This can also be used in many applications for the line coding purpose in order to remove the error occurrence during the transmission of data from one place to another.*

Keywords- NRZI, SPARTAN 3A, FPGA, XILINX, VHDL.

I. INTRODUCTION

NRZI is a method of mapping a binary signal to a physical signal for transmission over some transmission media. The two level NRZI signal has a transition at a clock boundary if the bit being transmitted is a logical 1, and does not have a transition if the bit being transmitted is a logical 0. NRZI may have a long series of 0s or 1s, resulting in clock recovery difficulties. It is a datarecording and transmission method that ensure clock synchronization. This is especially helpful in situation where bit stuffing is employed, the practice of adding bit to a data stream so it conforms to communication protocol. This added bits can create long string of similar bits, which is register to the receiver as single, unchanging voltage since clock is adjust on voltage changes large behind true time. It ensures that after zero bit appears the voltage will immediately switch to one bit voltage level. The voltage changes allow sending and receiving clock to synchronization. An additional encoding mechanism must be used to ensure clock recovery. Run-Length Limited (RLL) encoding, such as that used with magnetic disk storage device, is preferred over Universal Serial Bus (USB) bit stuffing, which often results in variable Data Transfer Rates (DTR).

II. OBJECTIVES

The aim of the project is to replace the existing digital line codes such as NRZ, RZ used for digital communication by the NRZI technique. It servers the purpose of providing error free communication and solves the problem of sending continuous HIGH or LOW signals. It also provides secure and reliable data transfer.use automatic hyphenation and check your spelling. Additionally, be sure your sentences are complete and that there is continuity within your paragraphs. Check the numbering of your graphics (figures and tables) and make sure that all appropriate references are included.

A. Existing System

TheNRZ is one of the most basic of coding scheme. In this method message signal does not return to zero after each bit frame. This means that the message exactly follows the digital data structure. For example, a long data string of “1”s will produce a long period in the message signal. Transitions only occur in the message when there is a logical bit change. This is a very easy method to implement on the encoding side but requires the data rate to be known exactly on the receiving side but requires the data to be known exactly on the receiving side in order to be decoded. Any mismatch in data clock timings will result in erroneous data that is only detectable with some error detection such as check sum or CRC. Also errors from the communication channel or interference will not be detected without some form of data integrity checks.RZ- It describes the line codes used in telecommunication system in which the signal drops to zero between each pulse. This takes place even if a number of consecutive 0’s or 1’s occur in the signal. The signal is self-clocking. This means that separate clock does not need to be sent alongside the signal, but suffers from using twice the bandwidth to achieve the same data rate as compared to non-return-to-zero (NRZ) format.

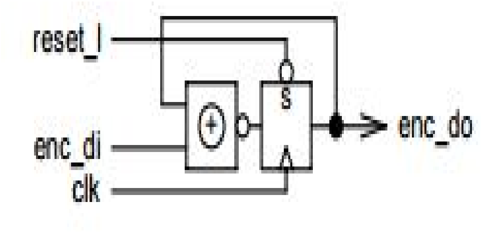
B. Advantage of proposed system.

It supports clock self-synchronization. It supports bit stuffing.it reduces the delay compared to other line coding techniques. It is majorly used for serial communication.

III. OPERATION

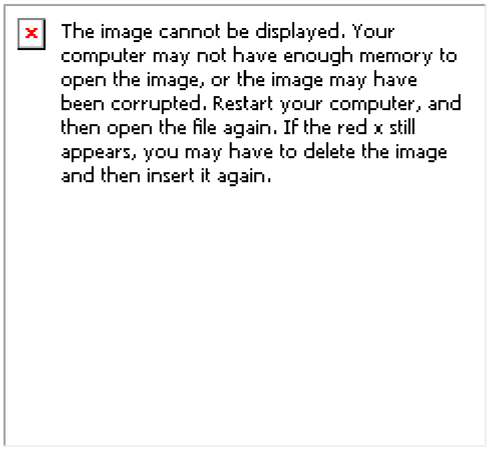
A. Encoder

The working of the encoding starts only when the clock signal is high and also the reset should be high at the SR flip flop. The transmission starts only when the bit is high and no transition occurs at low bit. As both the clock &reset is high the EX-OR block takes in its first input bite. Performs its EX-OR operation, inverses the bit and pass away to the SR flip flop.



Block diagram of Encoder

According to SR operation it will produce the output. Then it is again given as feedback to the EX-OR. The EX-OR would perform its operation and inverts it and again passes to the SR flip flop.



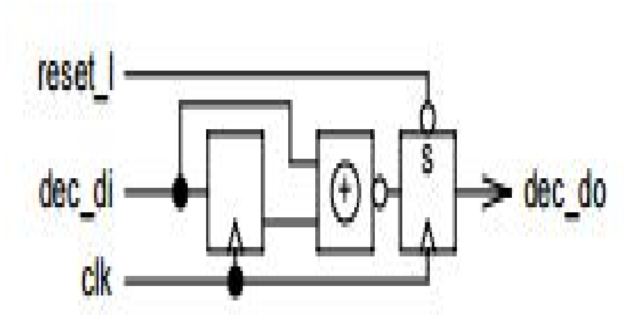
The image cannot be displayed. Your computer may not have enough memory to open the image, or the image may have been corrupted. Restart your computer, and then open the file again. If the red x still appears, you may have to delete the image and then insert it again.

The SR flip flop would produce the output at the end. The output of the encoder is given as the input to the decoder.

Inputs		Outputs		Action
S	R	Q	Q’	
0	0	Q_n	Q'_n	No change
0	1	0	1	Reset
1	0	1	0	Set
1	1	0	0	Undefined

B. Decoder

The output of the encoder is given as the input to the decoder. Then the decoder works occur at both the high & low signals. It is the advantage of this paper.



Block diagram of Decoder

As the previous paper are not clock synchronized it produce delay. As the D flip-flop takes the input with a delay it is passed to the EX-OR. The EX-OR performs its function inverts the bit & passes to SR. The SR produces the output. The output produced at the end of decoder would be same as the input fed to the encoder. Enc _di is an input to the NRZI encoder. This is an original serial data to be encoded. The NRZI encoder outputs enc_d0. This is serial data encoder is transmitter on serial transmission line and becomes an input of NRZI decoder. Dec_d0 is an output of the NRZI decoder. The serial data exactly same as the original serial data, Enc_di, must be reproduced after the decoding. Two bit delay happens between enc_di and dec_d0 to achieve stable digital encoding and decoding.

IV. SOFTWARE AND HARDWARE

A. Xilinx ISE

Xilinx ISE (Integrated Software Environment) is a software tool produced by Xilinx for synthesis and analysis of HDL designs, enabling the developer to synthesize (‘compile’) their designs, perform timing analysis, examine RTL diagrams, simulate a design’s reaction to different stimuli and configure the target device with the programmer.

SYNTHESIS AND SIMULATION

Synthesis is a process by which an abstract form of desired circuit behavior, typically registered transfer level (RTL) is turned into a design implementation in terms of logic gates. Common example of this process include synthesis of HDLs, including VHDL and Verilog.

Synthesis is the translation process from a description of a hardware device at higher abstraction level into an optimized implementation on a lower level abstraction. This process may be done by human or computer program. There is surge of incentive to program automatic synthesis programs for VLSI designs because

1. VLSI complexity has increased tremendously recently.
2. Demand for shorter and shorter design cycle (time to market).

There are two general categories of synthesis process

1. Behavior –to –structure
2. Structure –to-physical layout

We can also view the synthesis process in three different levels

1. Behavior synthesis
2. Logical synthesis
3. Physical synthesis

Simulation is the imitation of the operation of a real-world process or system over time. The act of simulating something first requires that a model be developed. This model represent the key characteristics or behaviors / function of the selected physical or abstract system or process. Simulation can be shown the eventually real effects of alternative conditions and courses of action. Simulation is also used when the real system cannot be engaged, because it may not be accessible, or it may be dangerous or unacceptable to engage, or it is being designed but not yet build, or it may be simply not exist.

FPGA DESIGN FLOW

Field-Programmable Gate Array (FPGA) is a device that has numerous gate (switch) arrays and can be programmed on-board through dedicated Joint Test Action Group (JTAG) or on-board device or using remote system through Peripheral Component Interconnect Express (PCIF), Ethernet, etc. FPGAs are based on Static Random Access Memory (SRAM). The contents of the Memory FPGA Arrays one‘s the power is turned off.

XILINX DEVICE PROGRAMMING

PROGRAM YOUR XILINX DEVICE AS FOLLOW

1. Create a programming file to program your FPGA.
2. Generate a PROM or ACE file for debugging or to download to your device. Optionally, create a JTAG file.

B. SPARTAN 3A FPGA

To define the behavior of the FPGA, the user provides a hardware description language (HDL) or a schematic design. The HDL form is more suited to work with large structures because it's possible to just specify them numerically rather than having to draw every piece by hand. However, schematic entry can allow for easier visualization of a design.

Then, using an electronic design automation tool, a technology-mapped net list is generated. The netlist can then be fitted to the actual FPGA architecture using a process called place-and-route, usually performed by the FPGA Company’s proprietary place-and-route software. The user will validate the map, place and route results via timing analysis, simulation, and other verification methodologies. Once the design and validation process is complete, the binary file generated (also using the FPGA company's proprietary software) is used to reconfigure the FPGA. This file is transferred to the FPGA/CPLD via a serial interface (JTAG) or to an external memory device like an EEPROM.

ADVANTAGE

The advantages of the field programmable gate array are as follows:

- Ability to re-program in the field to fix bugs.
- Shorter time to market.
- Lower non-recurring engineering costs.

ARCHITECTURE

The most common FPGA architecture consists of an array of logic blocks (called Configurable Logic Block, CLB, or Logic Array Block, LAB, depending on vendor), I/O pads, and routing channels. Generally, all the routing channels have the same width (number of wires). Multiple I/O pads may fit into the height of one row or the width of one column in the array.

An application circuit must be mapped into an FPGA with adequate resources. While the number of CLBs/LABs and I/Os required is easily determined from the design, the number of routing tracks needed may vary considerably even among designs with the same amount of logic. For example, a crossbar switch requires much more routing than a systolic array with the same gate count. Since unused routing tracks increase the cost (and decrease the performance) of the part without providing any benefit, FPGA manufacturers try to provide just enough tracks so that most designs that will fit in terms of Lookup tables (LUTs) and I/O can be routed. This is determined by estimates such as those derived from Rent's rule or by experiments with existing designs.

In general, a logic block (CLB or LAB) consists of a few logical cells (called ALM, LE, Slice etc.). A typical cell consists of a 4-input LUT, a Full adder (FA) and a D-type flip-flop, as shown below. The LUTs are in this Figure split into two 3-input LUTs. In normal mode those are combined into a 4-input LUT through the left multiplexer. In arithmetic mode, their outputs are fed to the FA. The selection of mode is programmed into the middle multiplexer. The output can be either synchronous or asynchronous, depending on the programming of the multiplexer to the right, in the Figure example. In practice, entire or parts of the FA are put as functions into the LUTs in order to save space.

Illustration of a Logic Cell

HARD BLOCKS

Modern FPGA families expand upon the above capabilities to include higher level functionality fixed into the silicon. Having these common functions embedded into the silicon reduces the area required and gives those functions increased speed compared to building them from primitives. Examples of these include multipliers, generic DSP blocks, embedded processors, high speed I/O logic and embedded memories.

Higher-end FPGAs can contain high speed multi-gigabit transceivers and hard IP cores such as processor cores, Ethernet MACs, PCI/PCI Express controllers, and external memory controllers. These cores exist alongside the programmable fabric, but they are built out of transistors instead of LUTs so they have ASIC level performance and power consumption while not consuming a significant amount of fabric resources, leaving more of the fabric free for the application-specific logic. The multi-gigabit transceivers also contain high performance analog input and output circuitry along with high-speed serializes and de-serializes, components which cannot be built out of LUTs. Higher-level PHY layer functionality such as line coding may or may not be implemented alongside serializes and de-serializes in hard logic, depending on the FPGA.

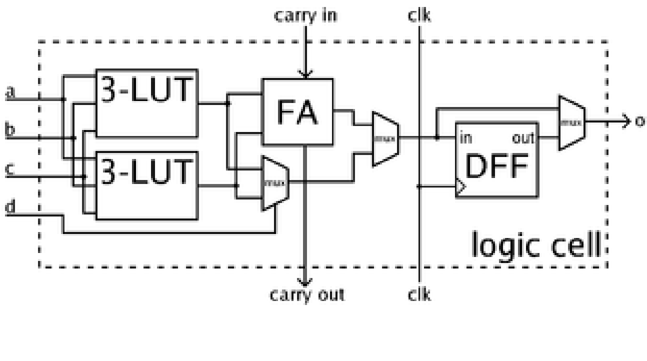
CLOCKING

Most of the circuitry built inside of an FPGA is synchronous circuitry that requires a clock signal. FPGAs contain dedicated global and regional routing networks for clock and reset so they can be delivered with minimal skew. Also FPGAs generally contain analog PLL and/or DLL components to synthesize new clock frequencies as well as attenuate jitter. Complex designs can use multiple clocks with different frequency and phase relationships, each forming separate clock domains. These clock signals can be generated locally by an oscillator or they can be recovered from a high speed serial data stream. Care must be taken when building clock domain crossing circuitry to avoid stability. FPGAs generally contain block RAMs that are capable of working as dual port RAMs with different clocks, aiding in the construction of building FIFOs and dual port buffers.

3D ARCHITECTURES

To shrink the size and power consumption of FPGAs, vendors such as Tabula and Xilinx have introduced new 3D or stack. Architectures. Following the introduction of its 28 nm 7-series FPGAs, Xilinx revealed that several of the highest-density parts in those FPGA product lines will be constructed using multiple dies in one package, employing technology developed for 3D construction and stacked-die assemblies.

Xilinx's approach stacks several (three or four) active FPGA die side-by-side on a silicon interposer a single piece of silicon that carries passive interconnect. The multi-die construction also allows different parts of the FPGA to be created with different process technologies, as the process requirements are different between the FPGA fabric itself and the very high speed 28 bits serial transceivers. An FPGA built in this way is called a heterogeneous FPGA.



Altera's heterogeneous approach involves using a single monolithic FPGA die and connecting other die/technologies to the FPGA using Intel's embedded Multi-Die Interconnect Bridge (EMIB) technology.

SPARTON 3A KIT

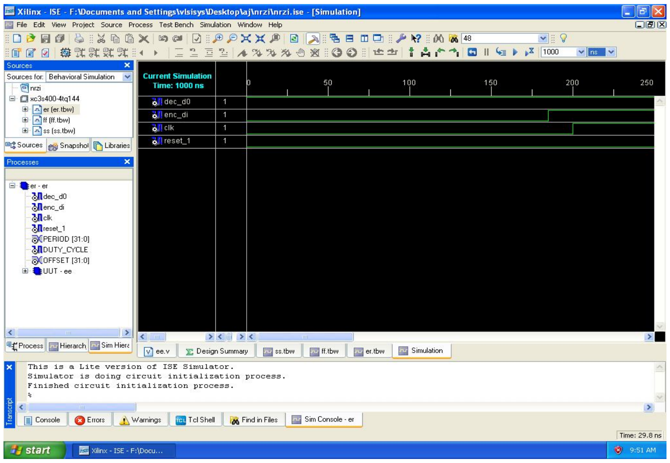
V. IMPLEMENTATION OF NRZI CODING THROUGH VHDL AND VERILOG HDL IN SPARTAN IN FPGA

The NRZI encoder and decoder operation is written in VHDL and Verilog. The source program is begin present which are later called in the main program. While executing the program and downloading it in the kit, it is sufficient to generate PROM file of the main program.

A. PROCEDURE FOR IMPLEMENTING NRZI ENCODER AND DECODER IN XILINX ISE

The following steps are followed to implement NRZI encoder and decoder in Xilinx ISE.

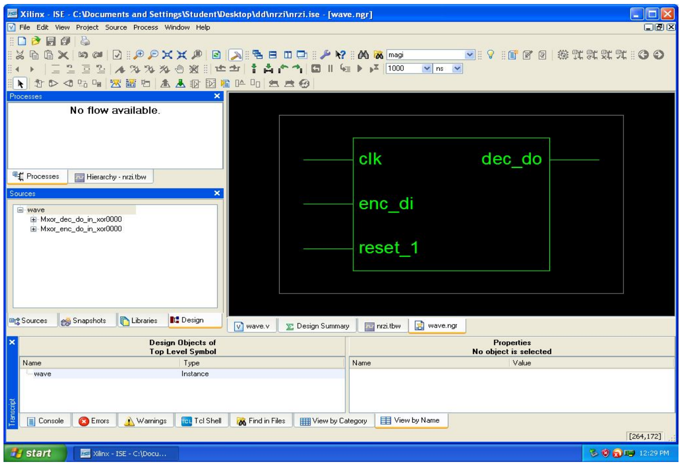
1. Open the Xilinx ISE.
2. Open FILE→NEW PROJECT.
3. Name the file.
4. Language is chosen at which we are going to work.
5. At add source input and output are declared.
6. Next FINISH.
7. At the program location the program is typed.
8. Then the program is saved.
9. At SYNTHESIS- XST→ CHECK SYNTAX.
10. At add new source→ TEST BENCH WAVEFORM is added.
11. Next FINISH.
12. Click ISE SIMULATOR→ SIMULATE BEHAVIOR MODEL.
13. Waveform is displayed.



```
This is a Lite version of ISE Simulator.
Simulator is doing circuit initialization process.
Finished circuit initialization process.
%
```

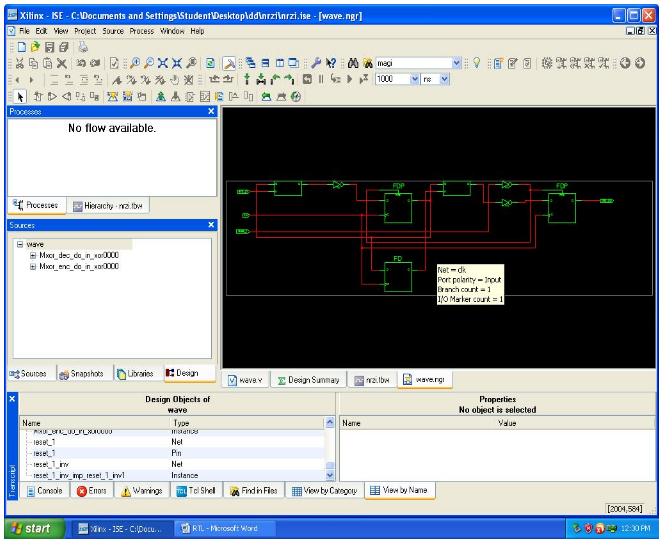
Test bench waveform

14. **SYNTHESIS-XST→**View RTL schematic.



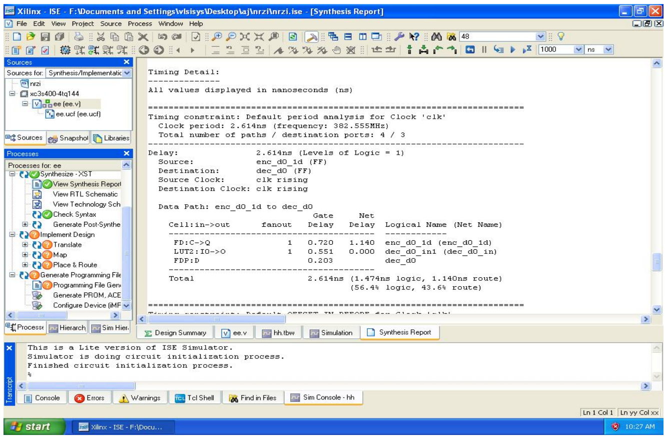
RTL schematic

15. On double clicking the RTL schematic the LUT is displayed.



Look up table

16. And again click on VIEW SYNTHESIS REPORT.



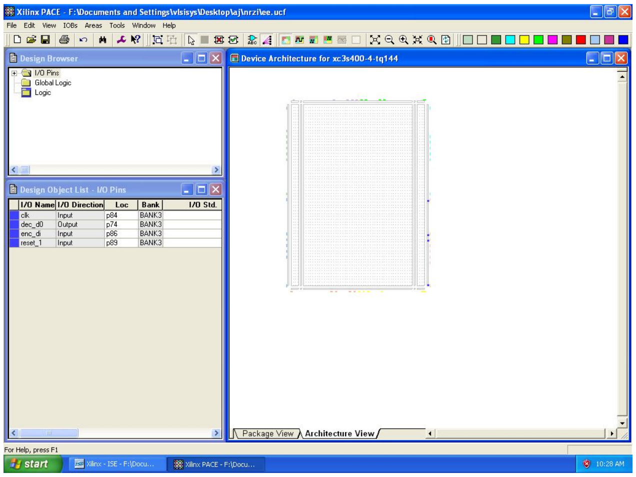
```
Timing Detail:
--------------
All values displayed in nanoseconds (ns)

=========================================================================
Timing constraint: Default period analysis for Clock 'clk'
  Clock period: 2.614ns (frequency: 382.555MHz)
  Total number of paths / destination ports: 4 / 3
-------------------------------------------------------------------------
Delay:               2.614ns (Levels of Logic = 1)
  Source:            enc_d0_1d (FF)
  Destination:       dec_d0 (FF)
  Source Clock:      clk rising
  Destination Clock: clk rising

  Data Path: enc_d0_1d to dec_d0
                                Gate     Net
    Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
    ----------------------------------------  ------------
     FD:C->Q               1   0.720   1.140  enc_d0_1d (enc_d0_1d)
     LUT2:I0->O            1   0.551   0.000  dec_d0_in1 (dec_d0_in)
     FDP:D                     0.203          dec_d0
    ----------------------------------------
    Total                      2.614ns (1.474ns logic, 1.140ns route)
                                       (56.4% logic, 43.6% route)

=========================================================================
```

```
This is a Lite version of ISE Simulator.
Simulator is doing circuit initialization process.
Finished circuit initialization process.
%
```

Synthesis report

17. Click **USER CONSTAINT→ ASSIGN PIN PACKAGE.**

Input and the output are declared for the ports.



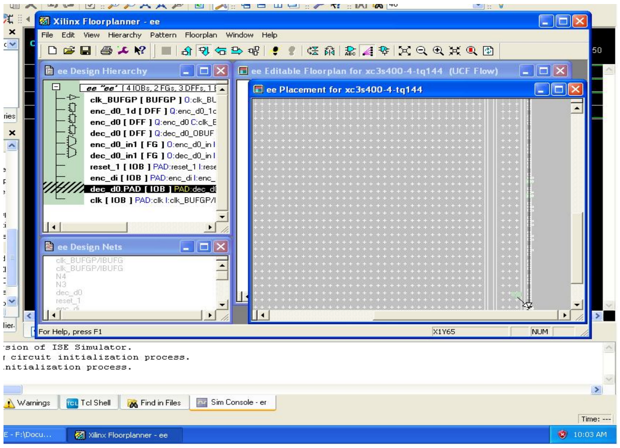
Design Object List - I/O Pins

	I/O Name	I/O Direction	Loc	Bank	I/O Std.
	clk	Input	p84	BANK3	
	dec_d0	Output	p74	BANK3	
	enc_di	Input	p86	BANK3	
	reset_1	Input	p89	BANK3	

Xilinx PACE - F:\Documents and Settings\vlsisys\Desktop\aj\ee.ucf
File Edit View IOBs Areas Tools Window Help
Design Browser
I/O Pins
Global Logic
Logic
Device Architecture for xc3s400-4-tq144
Package View
Architecture View
For Help, press F1
start
10:28 AM

Pin Package Assignment

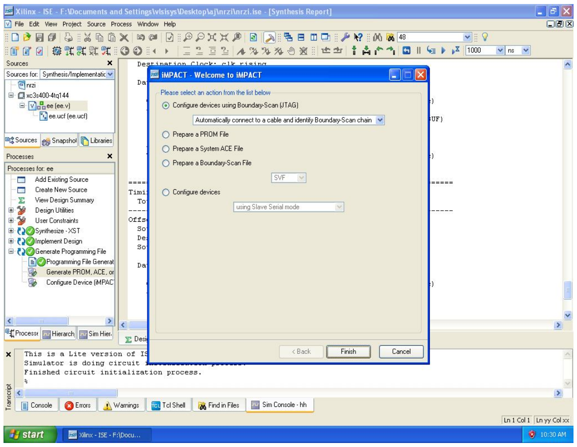
18. Click **IMPLEMENT DESIGN→PLACE AND ROUTE→**then double click the **VIEWED IT PLACED DESIGN**.



Xilinx Floorplanner - ee
File Edit View Hierarchy Pattern Floorplan Window Help
ee Design Hierarchy
ee "ee" [4 IOBs, 2 FGs, 3 DFFs, 1 E
clk_BUFGP [BUFGP] O:clk_BU
enc_d0_1d [DFF] Q:enc_d0_1c
enc_d0 [DFF] Q:enc_d0 C:clk_E
dec_d0 [DFF] Q:dec_d0_OBUF
enc_d0_in1 [FG] O:enc_d0_in I
dec_d0_in1 [FG] O:dec_d0_in I
reset_1 [IOB] PAD:reset_1 I:rese
enc_di [IOB] PAD:enc_di I:enc_
dec_d0.PAD [IOB] PAD:dec_d
clk [IOB] PAD:clk I:clk_BUFGP/I
ee Design Nets
clk_BUFGP/IBUFG
clk_BUFGP/IBUFG
N4
N3
dec_d0
reset_1
ee Editable Floorplan for xc3s400-4-tq144 (UCF Flow)
ee Placement for xc3s400-4-tq144
For Help, press F1
X1Y65
NUM
Warnings
Tcl Shell
Find in Files
Sim Console - er
Time: ---
Xilinx Floorplanner - ee
10:03 AM

Place and Route

19. Click **GENERATE PROGRAMING FILE→GENERATE PROM FILE** and click **FINISH**.



iMPACT - Welcome to iMPACT

Please select an action from the list below

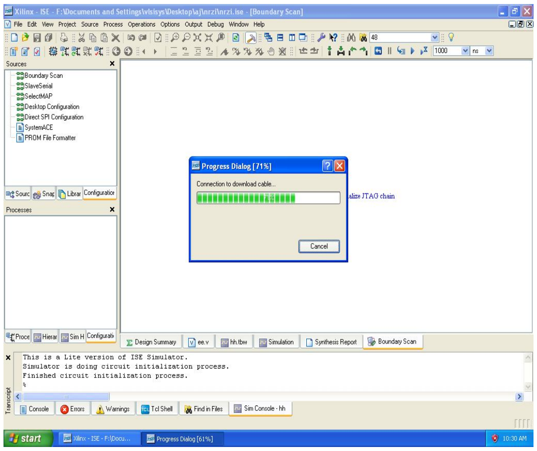
- ● Configure devices using Boundary-Scan (JTAG)
  - Automatically connect to a cable and identify Boundary-Scan chain
- ○ Prepare a PROM File
- ○ Prepare a System ACE File
- ○ Prepare a Boundary-Scan File
  - SVF
- ○ Configure devices
  - using Slave Serial mode

< Back | Finish | Cancel

```
This is a Lite version of ISE Simulator.
Simulator is doing circuit initialization process.
Finished circuit initialization process.
%
```

IMPACT Dialog box

20. Then progress dialog box will appear and show 100% of its connection setup.



Progress Dialog [71%]

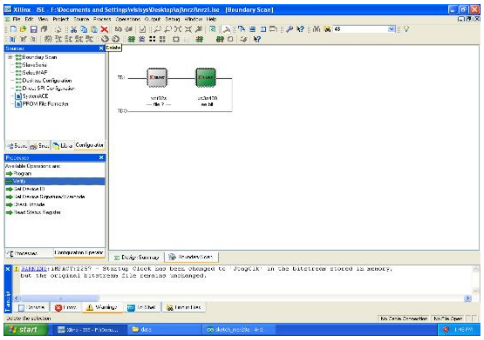
Connection to download cable...

Cancel

```
This is a Lite version of ISE Simulator.
Simulator is doing circuit initialization process.
Finished circuit initialization process.
%
```

Progress dialog box

21. Then boundary scan will display, XILINX block is double clicked and add file is added.

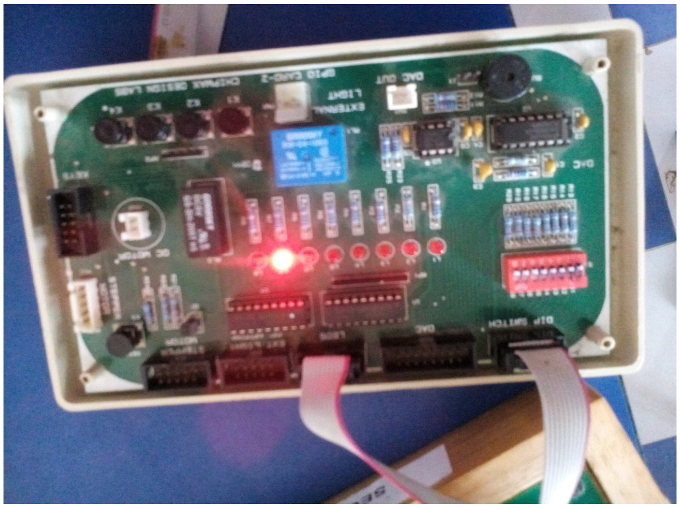


```
WARNING:iMPACT:2257 - Startup Clock has been changed to 'JtagClk' in the bitstream stored in memory,
but the original bitstream file remains unchanged.
```

Boundary Scan Dialog Box

22.Then click on **bypass** at **XILINX** block and the signal would get passed to the next block.
23.Then second block is double clicked, a dialog box will appear then click **open pass**.
24. Then click on **PROCESS**→click on **verify**

25. Finally the output would be generated.

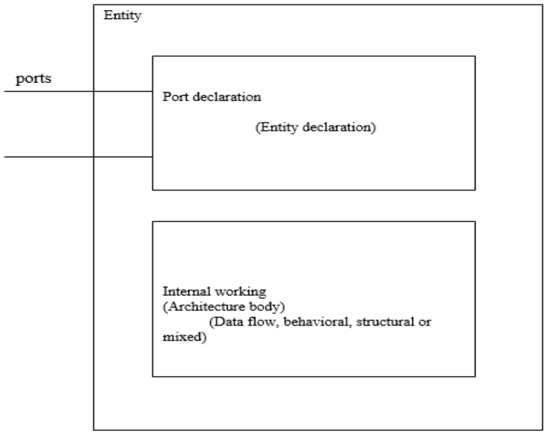


Final output

VI. VERILOG HDL

VHDL stands for very high–speed integrated circuit hardware description language, which is one of the programming languages used to model a digital system by dataflow, behavioral and structural style of modeling. This language was first introduced in 1981 for the Department Of Defense (DOD) under the VHSIC program. In 1983 IBM, Texas instruments and Intemetrics started to develop this language. In 1985 VHDL 7.2 version was released. In 1987 IEEE standardized the language. VHDL can also be used as a general purpose parallel programming language

A. STRUCTURE OF ENTITY



VII. CONCLUSION

"VHDL implementation of advanced NRZI encoder and decoder" gives an easy and less expensive way to implement NRZI coding system. First the different components that constitute the coder have been designed with required inputs, outputs and control signals. The interface between the designed components have been formulated. This was then realized in VHDL language using MODEL SIM software. The design was thoroughly simulated to test all the instructions in VHDL simulator software. As the coding system has high accuracy, the system gives more efficiency in sending data sending. This coding system is a universally accepted digital coding system. Also it provides more security for data transmission.

REFERENCES

[1] Digital Integrated Circuits-Janm. Rabey, AnanthaChandrasekaran

[2] VLSI Design flow- Neil.H.E.Weste, David Harris, AyanBanarjee

[3] PeterJ.Winzer , Associate member, IEEE, Alan H.Gnauck, Senior member, IEEE , Greg Raybon . member, IEEE,S.ChandraSekar, Fellow, IEEE,Yikisu, member,IEEE and Juerglleuthold member IEEE 2003-" 40-Gb/s Return-to-Zero Alternate-Mark-Inversion (RZ-AMI) Transmission Over 2000 km ".

[4] Spiros Mikroulis, Hercules Simos , Eugenia Roditi , Aristides Chipouras and Dimitris Syvridis2006-" 40-Gb/s NRZ and RZ Operation of an All-Optical AND Logic Gate Based on a Passive InGaAsP/InPMicroring Resonator ".

[5] Yu Zhang, Eniming Xu, Dexiu Huang, and Xinliang Zhang 2009-" All-Optical Format Conversion From RZ to NRZ Utilizing Microfiber Resonator".

[6] VLSI -Sahu, Tata MC Grew Hill Limited 2nd edition.

[7] Yu- Hsiangwen, student member, IEEE and Kai- Ming Feng, Member IEEE 2015-“On the Theory and Errata to “Data Transfer From RZ-OOK to RZ-BPSK by Polarization- Insensitive XPM in a Passive Birefringent Non-linear AlGaAs Waveguide”.