International Journal of Multidisciplinary Research and Technological Advancement (IJMRTA)

Volume 2 Issue 1, Year Jan-Feb 2024 *Available at www.ijmrta.in*

****

FPGA BASED IMPLEMENTATION NRZ, NRZI CODING TECHNIQUES

PUSHKAR JAWALE
Student, Electronics and Telecommunication Engineering department
Smt. Indira Gandhi College of Engineering, Navi Mumbai, Maharashtra, India

PRATIKSHA JADHAV
Student, Electronics and Telecommunication Engineering department
Smt. Indira Gandhi College of Engineering, Navi Mumbai, Maharashtra, India

DNYANESH DHAVADE
Student, Department of Computer Technology
K.K. Wagh Polytechnic, Nashik, Maharashtra, India

RAJ VICHARE
Student, Electronics and Telecommunication Engineering department
Smt. Indira Gandhi College of Engineering, Navi Mumbai, Maharashtra, India

****

ABSTRACT:

In this paper, we will look at how to use VHDL on the Xilinx or EDA Playground FPGA platform for online simulators to implement different line coding schemes. These schemes can help with data security, optimization of area, and efficient digital communication in different channel environments. Line codes are selected based on a number of factors, including the availability of DC level, power spectrum density, bandwidth needed, bit error rate (BER) performance, simplicity of clock signal recovery, and intrinsic error detection. The format for C Mark Inversion. Users may choose from a variety of line encoding methods or set various conditions for line coding techniques based on their needs by using the choice pin that is pressed into the chip. Xilinx or EDA Playground design tools are used for modeling and simulation of different line codes, while Spartan-6 FPGA or Tiny FPGA boards are used for hardware abstraction.

Keywords – VHDL, EDA, Xilinx, FPGA, DC Level, Spartan - 6

I. INTRODUCTION

To implement line coding, one must first determine the characteristics of the physical channel and the receiving equipment in order to create an amplitude-and time-discrete signal that ideally

represents the digital information to be sent. Researchers have been utilizing a variety of algorithms since the advent of VLSI technology in an effort to minimize propagation delay, optimize area, and decrease power consumption.

A variety of line encoding techniques are used, including Unipolar RZ and NRZ, Polar RZ and NRZI Coded, and others in A fundamental component of every communication system, line coding converts binary data into a series of voltage and current pulses that may be sent across physical medium such as optical fiber, coaxial cable, etc.

Typical spectrum properties of a pulse train are supplied by digital baseband signals. Among line codes, "return to zero" is by far the most used. They are all stored in FPGA in polar encoded or unipolar format. Considerations such as DC level, PSD (power spectrum density), bandwidth needs, bit error rate performance, simplicity of clock signal recovery, and intrinsic detecting properties determine the line codes to use.

II. LITERATURE SURVEY

These methods use the NRZ, NRZ-I, and RZ pulse forms to reduce inter symbol interference (ISI) by preventing successive pulse distortion and overlapping.

In a study published in April 2002 by R. S. Kaler et al. in Elsevier Optics Communication, the authors aimed to develop and apply an FPGA-based coding technique. The study also compared pre-, post-, and symmetrical-dispersion compensation schemes for 10Gb/s NRZ links using standard and SPDF fibers.

As stated in this study by Ajay K. Sharma (Oct 2001), the data is sent to the computer to activate the bit that receives signals. In order to fix Chromatic Dispersion, it employs an accurate signal at the receiver. Using a fiber optic system with a power level of 40 Gbps and a variety of Dispersion Compensation bit-error coding algorithms.

A study conducted by Tatsuya Kobayashi et al. in 1999 examined the use of digital baseband signals in NRZ-L and other Manchester line coding applications, such as providing specific spectral characteristics of pulse trains and validating and implementing the logic of various line coding techniques. The researchers also compared the performance degradation of DPSK and WDM transmission over transoceanic distances caused by nonlinear phase.

There are strong tools available to design field programmable gate arrays (FPGAs), which are single chips that have about 10 million logic gates and 10 million bits of memory (Carl R. Davidson et al., 2015). You, the programmer, have control over both the memory bits and the logical gates, unlike microprocessors. The steps involved in creating an FPGA design using programming are detailed in this article. The bits signals we have let us to communicate data from one computer to another with relative ease.

Linn F. Mollen Auer and colleagues A comparison of return-to-zero differential phase-shift keying, ON-OFF keying, and other unipolar and non-unipolar methods used in long-haul dispersion-controlled transmission, as well as their verification and implementation for digital output, demonstrates how simple it may be to obtain messages precisely.

Erichi Shibano, Kazuya Kilhida, and others The experimental comparison between the microprocessor and the field-programmable gate array (FPGA) reveals that the latter requires the Verilog or VHDL language and has a 100 MHz clock, while the former supports a 1 MHz clock, DPSK and QPSK in long-haul transmission, and signals with a data rate of 10 Gb/s. A few examples of line coding methods include the following: Unipolar NRZ, Unipolar RZ, Polar NRZ, Polar RZ, the Manchester format, Polar NRZ-I and NRZ-L, and a few variations on the biphasic and differential Manchester formats. With several levels of transmission.

We looked at a lot of different coding methods. Although these kinds of od=f coding strategies aren't exactly user-friendly, we've discovered that they're already part of our everyday lives. We are providing a selection of significant, practical, and user-friendly coding strategies for multi-level functionality, along with some of the implementation and verifications indicated in the literature review.

III. METHODOLOGY

Nowadays, we have a plethora of line coding schemes to choose from. In order to get some insight into the created project, several coding strategies were studied. The concept still has to be derived from the literature study, even if the methods used currently are more modern and fit with the newest technology.

Here we see the use of digital signals, which allow us to get the right signals to clarify our conditions and acquire the desired result; this output is often in the form of numbers or binary code, but thanks to technological advancements, we can transform this code into a digital format that is easy to understand. The "real world" that the FPGA will be linked to may be effectively simulated using the robust high-level structures provided by the Verilog language, which was originally developed for logic simulation.

Neither a gate-level netlist nor an FPGA logic structure mapping will be performed on the constructions. The testbench design is simpler since just a portion of the Verilog language may be synthesized; nevertheless, programming FPGAs is inherently more complicated. You may classify line coding schemes into the following general groups: -- Non-Return to Zero (Unipolar NRZ) Positive or negative voltage may be used to indicate unpolar mean bits. A high voltage level represents binary 1 in this encoded format, whereas a zero-voltage level represents binary 0 for the complete bit interval (Tb).

Universal Zero Return (RZ) In this encoded form, the first half of the bit interval ($0<t<Tb/2$) is represented by a high voltage (positive level), and the second half of the bit interval ($Tb/2<t<Tb$) is when the voltage returns to zero. For a complete bit interval (Tb), a voltage level of zero

represents binary 0. Non-Return to Zero Zone (NRZ) Polar The fact that bits may be represented by either positive or negative voltage levels is known as polarity. Bit 0 is represented by a low voltage (negative) level for the complete bit period (Tb), while high voltage (positive) level represents binary 1.

In this encoded format, the first half of the bit interval (0<t<Tb/2) is represented by a high voltage (positive) level, and the second half of the bit interval (Tb/2<t<Tb) is represented by a low voltage (negative) level. But 0 is represented by a low voltage (negative) level, and the second half of the bit interval (Tb/2<t<Tb) is also a return to zero. Format for Manchester Encoding For the first half of the bit interval (0<t<Tb/2) and the second half of the bit interval (Tb/2<t<Tb), a high voltage (positive) level represents binary 1, whereas a low voltage (negative) level represents bit 0, in this encoded format.

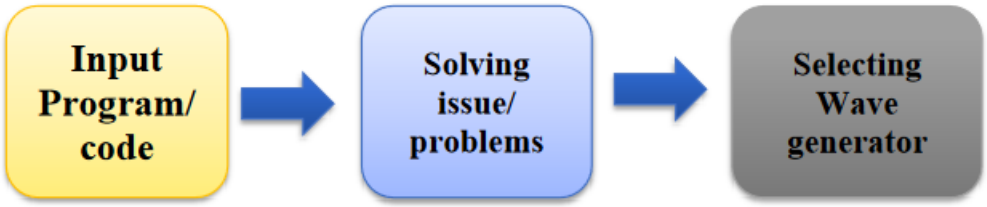

Figure. 3.1. Block Diagram of FPGA Based Implementation NRZ, NRZI Coding Techniques

Switching Mark Inversion (AMI) The zero-voltage level represents bit 0 in this encoded format for the complete bit interval (Tb), whereas the high voltage level and low voltage level alternately represent bit 1 for the length of the bit (Tb). Encoded Format—Pseudo Ternary Here, a zero-voltage level represents binary 1, while a high-voltage level and a low-voltage level alternate for the complete bit period (Tb) to represent binary 0.

This is the way we're going to input the Verilog code for the program, or implement it. At the outset, launch code compiler, programmer runner, or an online compiler; next, enter the verilog code into the test bench for testing purposes; at the same time, generate the primary code for Design.sv. In order to process the verilog code, we need to insert the main code into design.sv. We next need to enter some information and save the program, similar to how (testbench + design = Libraries, compiler, run time, etc.). Once all the coding is finished, all we need to do to see the waveform is click the Open EP wave button.

In these projects, all we need to do is verify the EP waveform using the ideal verilog code, or we may use one of many other kinds to produce the waveform while drawing the circuit schematic.

The digital signals are checked using these coding techniques before they are sent to the receiver in binary code. To see the waveform as an output, we simply input the correct code into the

simulator. This code, which is in the form of verilog code, is perfect for checking these FPGA projects.

We can see the waveform at any moment after making modifications to the codes in the testbench, but before we do so, we need to fill up the second side of the simulator, which is the real design of the line code.

IV. RESULT

In this picture, we can see the result of the provided verilog code; the input to the code is (1 0 1 1 0 1 0 1 1 1), and the result allows us to examine the coding methods used (via the use of the waveform). Here is the waveform shown by the line code: Line coding using unipolar NRZ In this area, the input signal is a binary 1 that begins at 1 (amplitude) and goes all the way to 0 (line of 0). Coding of unipolar RZ lines In this area, the polar RZ line coding for 0 signals is the line of 0, and the input is in binary form 1, therefore the amplitude starts at 1. In this area, the signal begins at 0.5 (amplitude) for 1, and it drops below the zero line for 0. Code for the Polar NRZ line In this area, the input signal is binary, with 1 beginning at 0.5 (the amplitude) and 0 approaching the -1 line (not touching at -1) as can be seen. Coding for the Manchester line Here we can see that the signal for 1 in this location begins at 0.5 (amplitude) and for 0 it approaches the line for -1 (does not touch at -1).

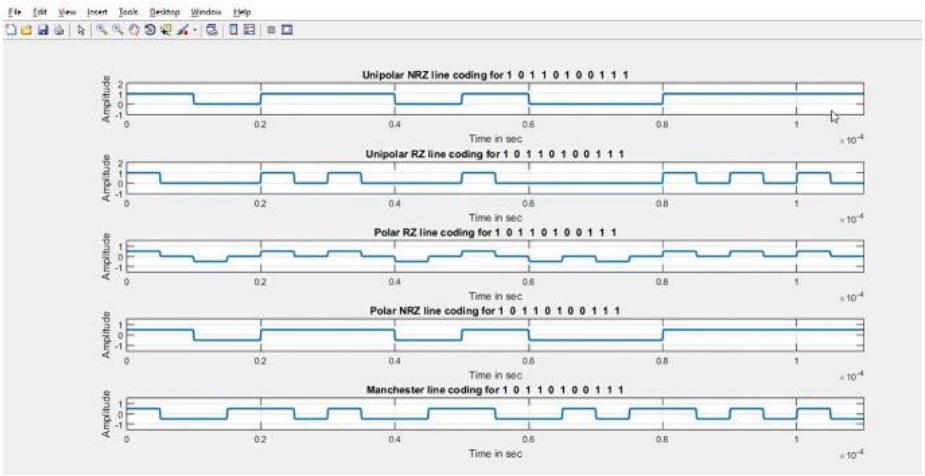

Figure 4.1. For (1 0 1 1 0 1 0 0 1 1 1) these inputs

Figure 2 shows the coded input (0 1 0 1 1 0 0 1) used to obtain a digital signal, and the waveform output allows us to verify the correctness of the coding. Here is the waveform shown by the line code:

In this area, we can see that the input signal, which is in binary 0 form, starts at 0 (amplitude) and moves to the 1 line for 1 in the case of unipolar NRZ line coding.

Coding of bipolar NRZ lines - The NRZI line coding allows us to observe that the input signal, in binary form, begins at -1 (amplitude) and goes to 1 (one) for 1 in this region.

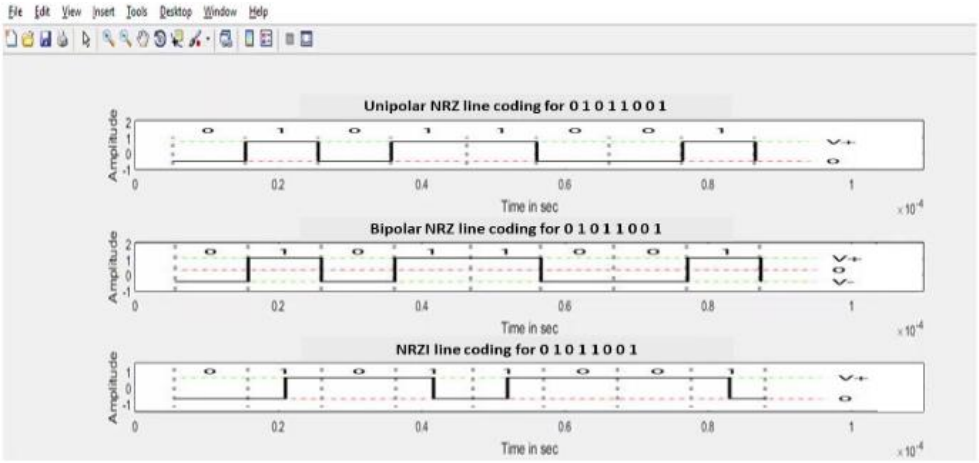

Figure 4.2. For (0 1 0 1 1 0 0 1) these inputs

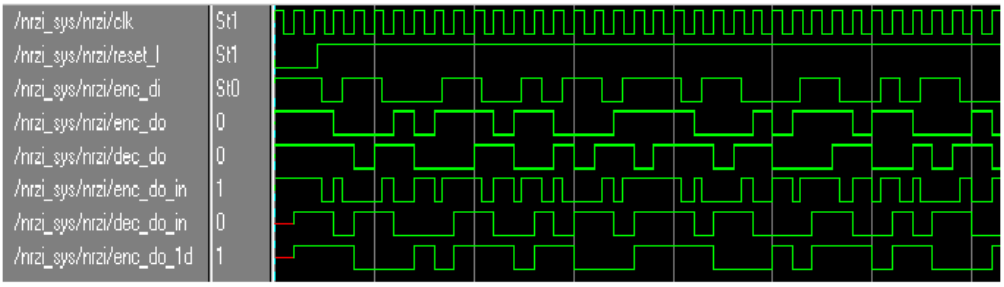

Figure 4.3. For (0 0 1 0 1) these inputs from verilog code

The "enc_do" is the output of the NRZI encoder. The input of the NRZI decoder is this serially encoded data that is sent via a serial transmission line.

The NRZI decoder puts out "dec_do" as its output. Reproducing the serial data, "enc_di," in an identical replica of the original serial data is required after decoding. The two-bit lag between "enc_di" and "dec_do" is necessary for reliable digital encoding and decoding.

V. CONCLUSSION

Using the EDA Playground online compiler or the Xilinx design tools and FPGA, this work implements and abstracts the hardware of several line coding schemes. Designed with simplicity, ease of implementation, and efficiency for all baseband signal encoding techniques in mind, this architecture is ideal for digital communication. We can see that part of the code starts with the binary code "0" by comparing the waveforms in the output to those of other line codes. What

we've learned about digital signals is that they employ binary code to communicate between devices. A string like "0011010" represents that binary code.

References:

[1]. Amrinder Kaur, Mandeep Singh, Balwinder Singh "VHDL Implementation of Universal Encoder for communication", ISP Journal of Electronics Engineering, Vol.1, Issue 2, ISSN December 2011.
[2]. David Alejandro Lopez-Ramirez, Mario Reyes-Ayala, Edgar Alejandro Andrade, Jose Alfredo Tirado-Mendez, "Educational prototype for line coding", proceeding of the6th WSEAS International Conference on Engineering Education, 2009.
[3]. J. Bhaskar, "A VHDL Primer". Third Edition, Person Education, Prentice Hall, 2008.
[4]. Chen Xinyuan, Zhou, Yu Jindong, "Design and Implementation of Manchester encoder", Journal of University of Electronics Science and Technology of June 2003.lletin, 30(1), 550-559.
[5]. L. Fu, Z.P. Ren, C.J. Liu: Journal of Modern electronic technology, Vol. 30 (2007) No.17, pp. 55-59 (In Chinese).
[6]. J. Dong, H.N. Dong and H. Chen: Journal of Wireless Communication Technology, (2010) No.3, pp. 5-8 (In Chinese).
[7]. W. Stallins: Data and Computer Communications (Eight edition) (Prentice Hall Press, America 2007).
[8]. H.G. Yang, J.B. Sun, W. Wang: Journal of Electronics Information Technology, Vol.32 (2010) No.3, pp. 714-723 (In Chinese).
[9]. U. Meyer-Baese: Digital Signal Processing with Field Programmable Gate Arrays (Third Edition) (Spinger Press, Germany 2007).
[10]. I.M Sobol, Y.L Levitan: Journal of Computers Mathematics with Applications, Vol.37 (2004), No. 4, pp. 33-40.
[11]. J.F. Li, M.G. Chai: Proceedings of the 2011 IEEE International Conference on Electronics, Communications and Control, (2011), pp.3642-3645