Design of CODEC Using VLSI Architecture for Efficient Communication

P.N.S. Krishna Mohan[1], Pavan Singarapu[2], P. Venkat Ram[3]

[1,2,3]Electronics and Communication Engineering, Gayatri Vidya Parishad College of Engineering (A)

Abstract:
Codec refers to coder-decoder, encoding and decoding play a vital role in networking, storage, data communication and wireless or radio communication systems. Thus, the hardware architecture of an encoding and decoding blocks becomes an attractive issue in designing. Very large-scale integration (VLSI) is one of the trending processes of integrating thousands of transistors together in a chip and its architecture provides better performance. When the VLSI architecture is involved in the coder design ,the system attains effective results in its parameters such as power consumption, speed and area occupied. In general, digital systems the Half Cycle Processing Model (HCPM) is totally different which is already existing for FMO/Manchester Codec. Through this model reduces number of transistor count and achieves 100% Hardware Utilization Rate, it induces one cycle latency between positive and negative half cycles. To resolve this problem, we are proposing a new coder design which is integrated with previous one to reduce the delay

1. INTRODUCTION:

1.1. Communication Process

The communication process is the guide toward realizing effective communication. It is through the communication process that the sharing of a common meaning between the sender and the receiver takes place. Individuals that follow the communication process will have the opportunity to become more productive in every aspect of their profession. Effective communication leads to understanding. The communication process is made up of four key components. Encoding, Medium of Transmission or Channel, Decoding and Feedback. There are also two other factors in the process, and those two factors are present in the form of the sender and the receiver. The communication process begins with the sender and ends with the receiver.

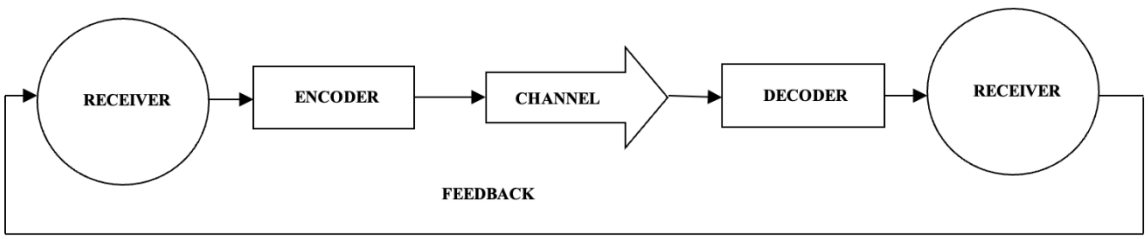

Fig.1. Encoding/Decoding model

Source/Encoder: The sender or source of a communication is the person or organization who has information to share with another person or group. It should be noted that the source can be an individual (e.g., salesperson or hired spokesperson) or a non-personal entity such as the corporation or organization itself. The sender's goal is to encode the message in such a manner to ensure that it will be understood by the receiver. The sender's experiences, attitudes, knowledge, skill, perceptions, and culture influence the message. "The written words,

spoken words, and nonverbal language selected are paramount in ensuring the receiver interprets the message as intended by the sender" (Burnett & Dollar, 1989).

Message: The encoding process leads to the development of a message that contains the information or meaning the source or sender hopes to convey. Messages can take a variety of forms and may include symbolic forms or signs. The message must be put into a transmittable form that is appropriate for the channel of communication being used.

Channel: The channel is the method or medium by which the communication travels from source or sender to receiver. At the broadest level, channels of communication exist as two types:

1. Personal Channels which involve direct interpersonal contact with target individuals or groups. For example, a salesperson serves as a personal channel of communication when delivering a sales presentation.
2. Non-personal channels are those which carry a message without involving interpersonal contact between sender and receiver. These channels are often referred to as the mass media as messages transmitted through them are sent to many individuals at one time. The two major categories of non-personal channels are print and broadcast media.

Receiver/Decoder: The receiver is the person(s) with whom the sender shares thoughts or information. After the appropriate channel or channels are selected, the message enters the decoding stage of the communication process. Decoding is conducted by the receiver. Once the message is received and examined, the stimulus is sent to the brain for interpreting, to assign some type of meaning to it. It is this processing stage that constitutes decoding.

Feedback: Response refers to the reaction the receiver has after seeing, hearing and/or reading the message. These responses can range from non-observable actions such as storing information in memory to taking immediate actions such as ordering a product seen in a direct response. Feedback is the part of the receiver's response that is communicated back to the sender and takes a variety of forms. Feedback provides the sender with a way of monitoring how the message is being decoded and received by the target audience. Feedback is the final link in the chain of the communication process. After receiving a message, the receiver responds in some way and signals that response to the sender. The signal may take the form of a spoken comment, a long sigh, a written message, a smile, or some other action. "Even a lack of response, is in a sense, a form of response" (Bovee&Thill, 1992).

1.2. CODES:

In communications and information processing, code is a system of rules to convert information such as a letter, word, sound, image, or gesture into another form or representation, sometimes shortened or secret, for communication through a channel or storage in a medium. An early example is the invention of language, which enabled a person, through speech, to communicate what he or she saw, heard, felt, or thought to others. But speech limits the range of communication to the distance a voice can carry and limits the audience to those present when the speech is uttered. The invention of writing, which converted spoken language into visual symbols, extended the range of communication across space and time

The process of encoding converts information from a source into symbols for communication or storage. Decoding is the reverse process, converting code symbols back into a form that the recipient of that understands time.

1.2.1. Types of Codes

A code is usually considered as an algorithm which uniquely represents symbols from some source alphabet, by encoded strings, which may be in some other target alphabet. An extension of the code for representing sequences of symbols over the source alphabet is obtained by concatenating the encoded strings. There are various types of codes used in communication systems. For example,

- Manchester Codes
- Flexible Macro-block (FMO) Codes
- Miller Codes
- Variable Length Codes
- Error Correcting Codes
- Binary Range Codes
- Turbo Codes
- Golay Codes
- Fault Secure Codes, etc.

Codes can be used for brevity. When telegraph messages were the state of the art in rapid long-distance communication, elaborate systems of commercial codes that encoded complete phrases into single mouths (commonly five-minute groups) were developed, so that telegraphers became conversant with such "words" as BYOXO ("Are you trying to weasel out of our deal?"), LIOUY ("Why do you not answer my question?"), BMULD ("You're a skunk!"), or AYYLU ("Not clearly coded, repeat more clearly.").

Code words were chosen for various reasons: length, pronounceability, etc. Meanings were chosen to fit perceived needs: commercial negotiations, military terms for military codes, diplomatic terms for diplomatic codes, all of the preceding for espionage codes. Codebooks and codebook publishers proliferated, including one run as a front for the American Black Chamber run by Herbert Yardley between the First and Second World Wars.

1.2.2 FMO Coding

FMO stands for Flexible Macroblock Ordering Coding. Also known as bi phase coding.

- **FMO Encoding:** The FMO code YFE is divided into two parts: YFP and YFN, as illustrated in Fig.2 The YFP and the YFN represent the FMO code at positive-cycle and negative-cycle respectively.

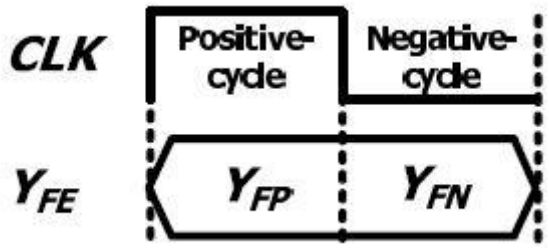

Fig.2 Structure of FMO Code

- Coding principals
  - Logic-0: Transition is allocated between YFP and YFN.
  - Logic- 1: Transition exists among every YFE.
  - A transition is allocated among each FMO code no matter what input is.
- The encoding procedure of FMO code is itemized as

$$YFP(t) = YFN(t-1)$$

$$YFN(t) = XE \oplus YFN(t-1)$$

Both YFP(t) and YFN(t) are three-variable Boolean functions of YFP(t−1) YFN(t−1), and XE. An example of FMO encoding is shown in Fig.3

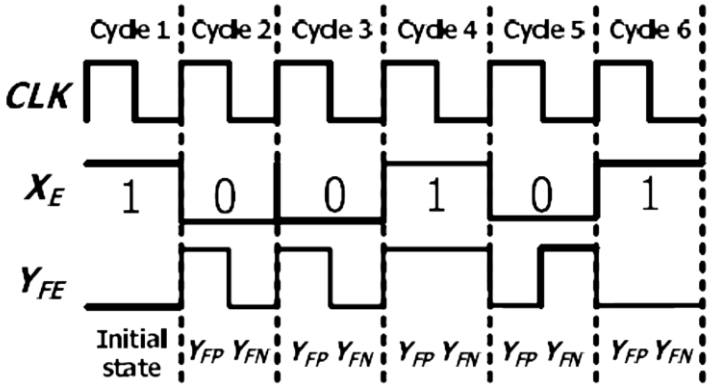

Fig.3. FMO Encoding

- **FMO Decoding:** For FMO decoding, the FMO code word to be decoded is represented by YFD, which also consists of YFP and YFN. The procedure of FMO decoding is given as follows
  - If a signal-transition exists between YFP and YFN, XD is interpreted as logic-0.
  - If no signal-transition exists between YFP and YFN, XD is interpreted as logic-1.
  - The logic value of XD only depends on whether there is a signal-transition between YFP and YFN.

1.2.3. Manchester Coding

Manchester coding is also called as Phase Coding. Manchester code is a line code in which the encoding of each data bit is either low then high, or high then low, for equal time. It is a Self-clocking signal with no DC component. As a result, electrical connections using a Manchester code are easily galvanically isolated.

Manchester Encoding and Decoding: The Manchester encoding is given as,

$YME = XE \oplus CLK$

Where, XE and YME stand for the binary data to be encoded and its corresponding Manchester code, respectively. The CLK is a clock signal. According to (1), YME must have a signal-transition in every cycle, whether XE is logic-1 orlogic-0. With this signal-transition, ac components are embedded into the YME, which can facilitate the synchronization in receiver. Similarly, the Manchester decoding is given as

$XD = YMD \oplus CLK$

Where, XD stands for the decoded binary data, and the YMD denotes a Manchester codeword to be decoded. Two examples are illustrated for Manchester encoding/decoding in Fig.4. and Fig.5. respectively. Both can be simply realized by a 2-input XOR gate. Manchester encoding/decoding has an identical logic function of 2-input XOR gate. From hardware perspective, this 2-input XOR gate can be fully reused by Manchester encoding and Manchester decoding. If Manchester encoding is adopted, XE is passed through MUX to XOR, and the output of XOR represents YME. If Manchester decoding is adopted, YMD is passed through MUX to XOR, and the output of XOR denotes XD.

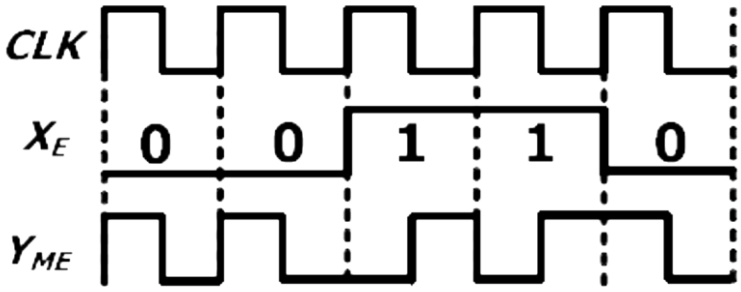

Fig.4. Manchester Encoding

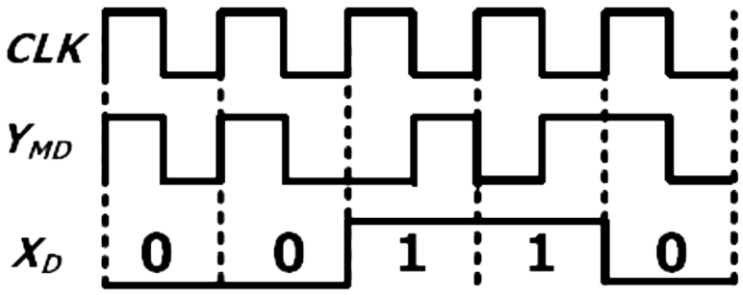

Fig.5. Manchester Decoding

The purpose of most of these codes was to save on cable costs. The use of data coding for data compression predates the computer era; an early example is the telegraph Morse code where more-frequently used characters have shorter representations. Techniques such as Huffman coding are now used by computer-based algorithms to compress large data files into a more compact form for storage or transmission.

2. LITERATURE SURVEY

In [1] P.Benabes, A.Gauthier, J.Ohman proposed "A Manchester code generator running at 1Ghz", designs a Manchester encoder architecture, and its target application is for optical communication. It utilizes both CMOS inverter and gated inverter to realize Manchester encoding. Based on literature [1], [2]A.Karagounis, A.Polyzos, B.Kotsos, N.Assimakis proposed "A 90nm Manchester Code Generator with CMOS switches running at 2.4Ghz and 5Ghz", In literature [2] gated inverter is replaced by NMOS device for a higher operation frequency. It is implemented in 90 nm CMOS technology, and its operation frequency is as high as 5 GHz. In [3]Wael M El-Medany proposed "FPGA Implementation of RDR Manchester and D-Manchester CODEC Design for Wireless Transceiver**",** The article [3] develops a hardware architecture to integrate both Manchester code and Differential Manchester (D-Manchester) code. In [4]Yu-Cherng Hung, Min-Ming Kuo, Chiou-Kou Tung, and Shao-Hui Shieh proposed "High-Speed CMOS Chip Design for Manchester and Miller Encoder", For radio frequency identification (RFID) applications, the article [4] develops a reused hardware architecture to perform both Manchester encoding and Miller encoding. This hardware architecture is realized in 0.35 um CMOS technology.

In [5]M Ayoub Khan, Manoj Sharma, Brahmanandha Prabhu R proposed "FSM based FMO and Miller encoder for UHFRFID Tag Emulator", It exploits the finite state machine (FSM) to develop a Manchester encoder architecture for ultra-high frequency (UHF) RFID Tag emulator. The implementation of this work is on FPGA prototyping system, where its maximum operation frequency reaches 256 MHz.

In [6]Juinn-Horng Deng, Feng-Chin Hsiao, and Yi-Hsin Lin proposed "Top Down Design of Joint MODEM and CODEC Detection Schemes for DSRC Coded-FSK Systems over High Mobility Fading Channels", It constructs hardware architectures to merge for DSRC applications. In [7]M.A.Khan, M.Sharma, and P.R.Brahmanandha, proposed "FSM based FMO and Miller encoder for UHF RFID tag emulator", In this paper the FSM-based design strategy is adopted to design FMO encoder and Miller encoder for UHF RFID Tag emulator.

In [8]F.N.Zhang, X.A.Wang, S.S.Yong, X.L.Shi, B.Liu, and Z.Y.Guo proposed "A multi-bit encoder and FMO/Miller decoder design for UHF RFID reader digital baseband", For UHF RFID reader applications, the literature [8] implements a FMO/Miller decoder with CMOS 0.18 um technology. In [9]N.F.B.Bautista and J.J.S.Marciano Jr. proposed "Enhanced FMO decoder for UHF passive RFID readers using duty cycle estimations", It develops a FMO decoder using duty cycle estimations to tolerate data rate deviation for UHF RIFD readers.

In [1]P.Benabes, A.Gauthier, J.Ohman proposed "A Manchester code generator running at 1 GHz". This paper deliberately gives the Manchester encoding technique using 0.35 us CMOS technology which gives an ideal

duty cycle of 50% at 1GHz. The design is shown in figure 6. As this design is a transistor based and the main advantage of this design is clock signal is running at the same frequency as the data. The encoder behaviour is critical during the second clock phase when the output signal is inverted. During the 50% duty cycle clock signal lead to generate the 38% duty cycle in worse case and 48% duty cycle for the best case. This code can help in design of photo receiver amplifier in the context of optical intercommunication.

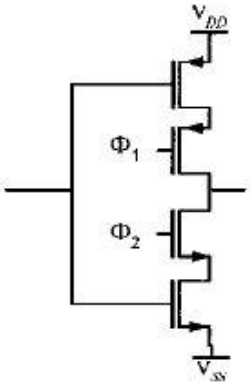

Fig.6. CMOS gated inverted design

In [2]A.Karagounis, A.Polyzos, B.Kotsos, N.Assimakis proposed "A 90nm Manchester Code Generator with CMOS switches running at 2.4GHz and 5GHz". This work presents a Manchester encoder with CMOS switches, running at the frequencies of 2.4 GHz and 5 GHz. The circuit is designed in such way that it behaves as a standard D flip flop and runs without the initial state. The encoder is shown in Fig.7. It is designed at transistor level and uses inverters and NMOS switches. The inverter delays are reduced by adjusting the transistor size at 2.4GHz frequency, using a 50% duty cycle clock signal lead to a generated output duty cycle of 40.3% for the worst-case process corner and 47.4% for the best case.At5 GHz frequency, using a 50% duty cycle clock signal lead to a generated output duty cycle of 39% for the worst-case process corner and 44.8% for the best case. The mean propagation time increases from 66ps to 84 ps. The mean propagation time increases from 138ps to 150ps. As the encoder behaviour is critical especially during the second clock phase when the output signal is inverted. These results are quite acceptable but not the best.

3. PROPOSED MODEL

The proposed VLSI architecture of FMO/Manchester codec is based on Half Cycle Partitioning Method (HCPM). The HCPM is a model of the hardware architecture for FMO/Manchester codec. It classifies FMO/Manchester encoding and decoding into two parts: positive-cycle signal and negative-cycle signal. Both are manipulated by positive-cycle logic and negative-cycle logic, respectively. With this classification, the function of FMO/Manchester codec can be transformed into the HCPM as illustrated in Fig.7. The HCPM model consists of three core techniques: HCLP, RBR, and BFR.

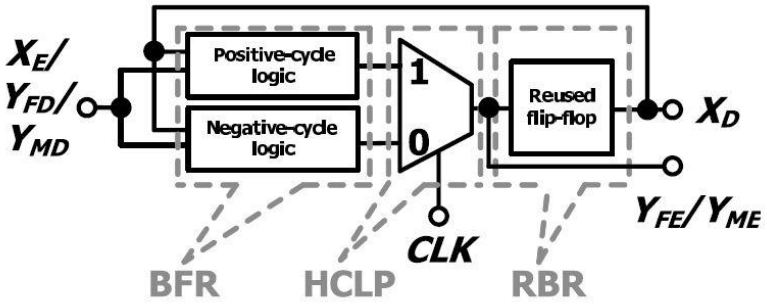

Fig.7. HCPM model of Codec

3.1. Half Cycle Logic Partitioning (HCLP):

The main purpose of HCLP technique is to differentiate negative cycle logic and positive cycle logic. Since FMO encoding is naturally conducted from positive cycle logic YFP(t) and negative cycle logic YFN(t), it is not in our discussion. Hence our discussion is mainly focused on the FMO decoding and Manchester codec. As we know, the Manchester codec can be realized by using a single 2 input XOR gate logic where XME, YMD are representing encoding and decoding of Manchester respectively with input X and clock. Where,

XME =X* CLK + X CLK*

But it can be transformed into 2 to 1 multiplexer where the CLK, X and X* are individually mapped to set X1, X2 as a result positive cycle logic and negative cycle logic of Manchester codec are XE*/YMD * and XE/YMD respectively.

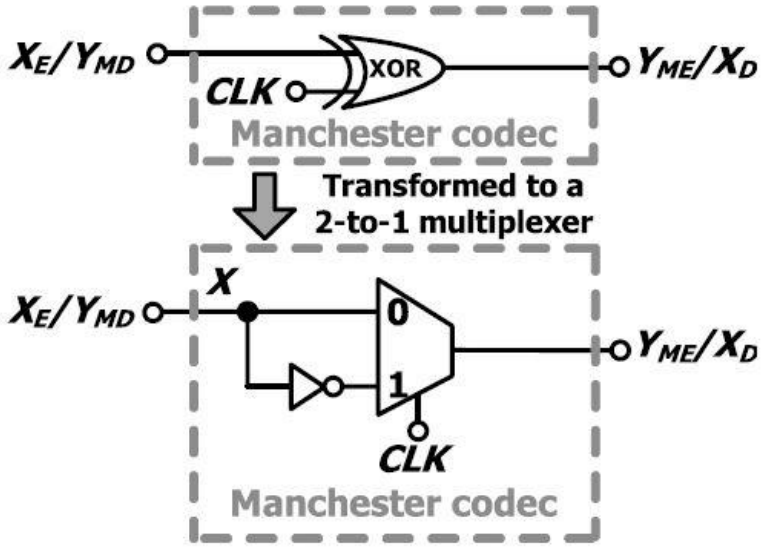

Fig.8. Manchester Codec with HCLP technique

XNOR function in FMO decoding can also be transformed to a 2×1 multiplexer even though clock not an input to it. This makes this data paths to fitted into positive and negative cycles logic. So, instead of transforming XNOR to 2×1 multiplexer as an alternative solution a pseudo multiplexer is introduced in FMO decoding. This pseudo multiplexer has two identical inputs. It is equivalent that both positive cycle logic and negative cycle logic are the same one by this both are fitted into the FMO decoding.

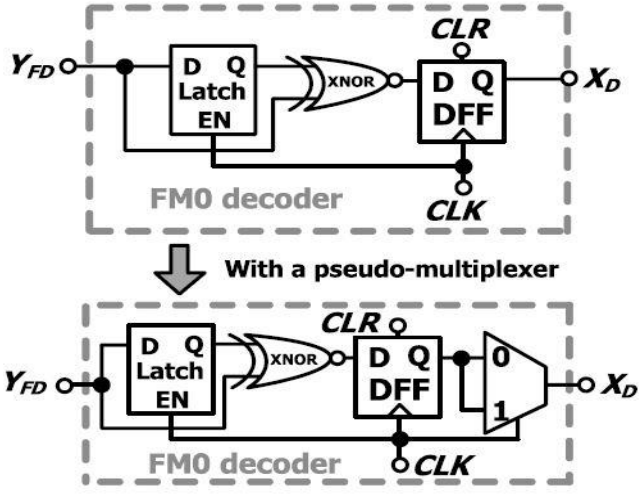

Fig.9. FMO Decoder with HCLP technique

3.2. Reused Based Retiming (RBR) Technique:

The purpose of RBR technique is to conduct a reused DFF from positive cycle logic and negative cycle logic in FMO/Manchester encoding and decoding. For FMO decoder, positive cycle logic and negative cycle logic are identical. So, the DFF can be reused with both. To fit the HCPM model, it is directly relocated backward after the multiplexer Fig.10

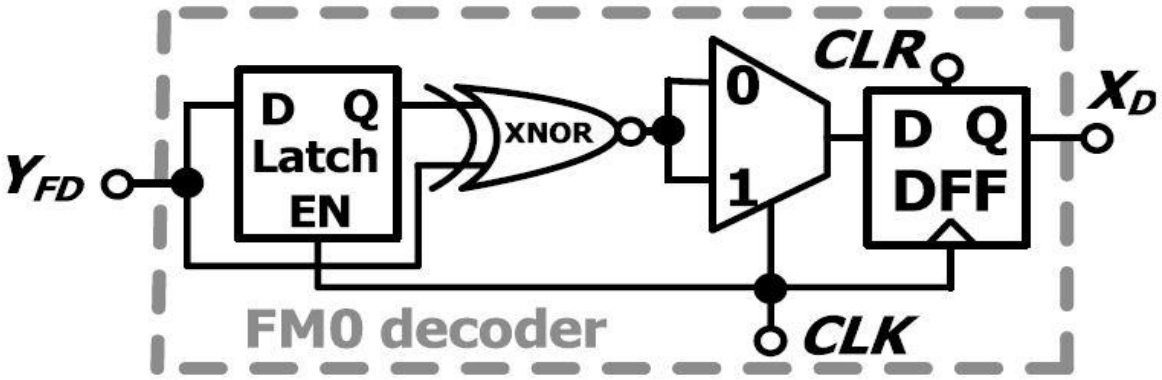

Fig.10. FMO Decoder with RBR Technique

For FMO encoder, to conduct a reused DFF from both positive cycle logic (YFP(t)) and negative cycle logic (YFN(t)), two conditions should be satisfied.

1) Both YFP(t) and YFN(t) are required to be stored into two individual DFF's.
2) The data paths from these two DFF's to a multiplexer are symmetrical.

From these two conditions, these two DFF's can be reduced to one, relocated after a multiplexer as a reused DFF.

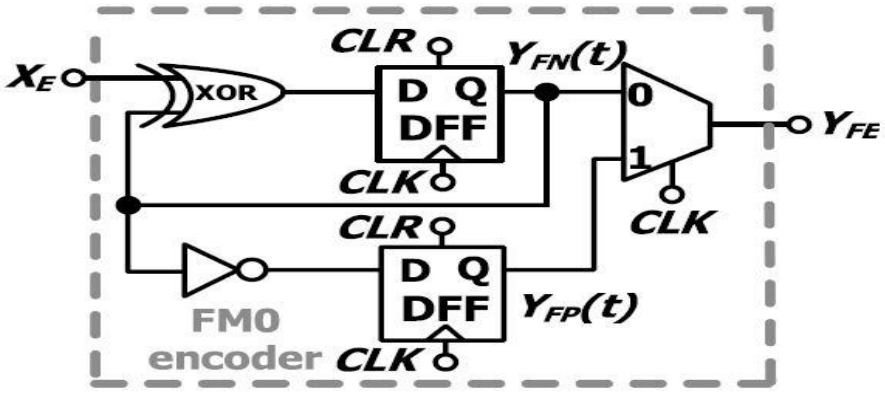

Fig.11. FMO Encoder with ne cycle latency for YFP(t), YFN(t)

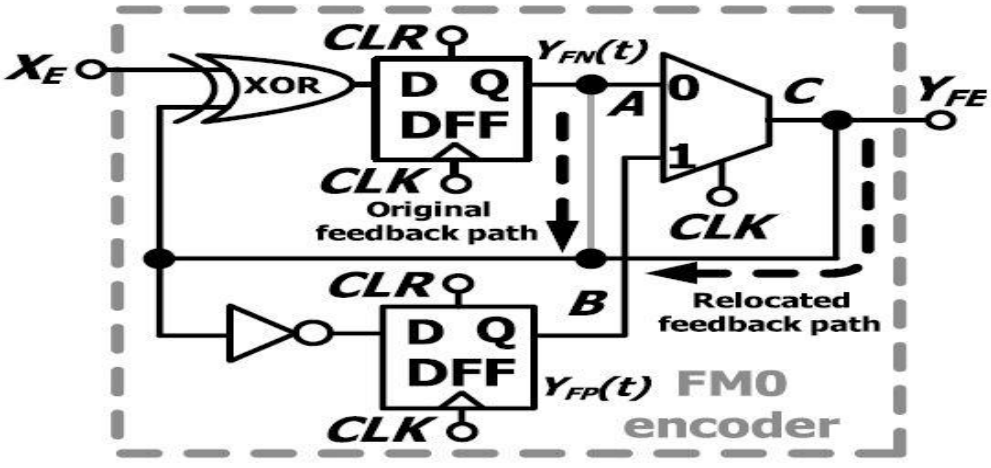

Fig.12. FMO Encoder with relocated feedback path

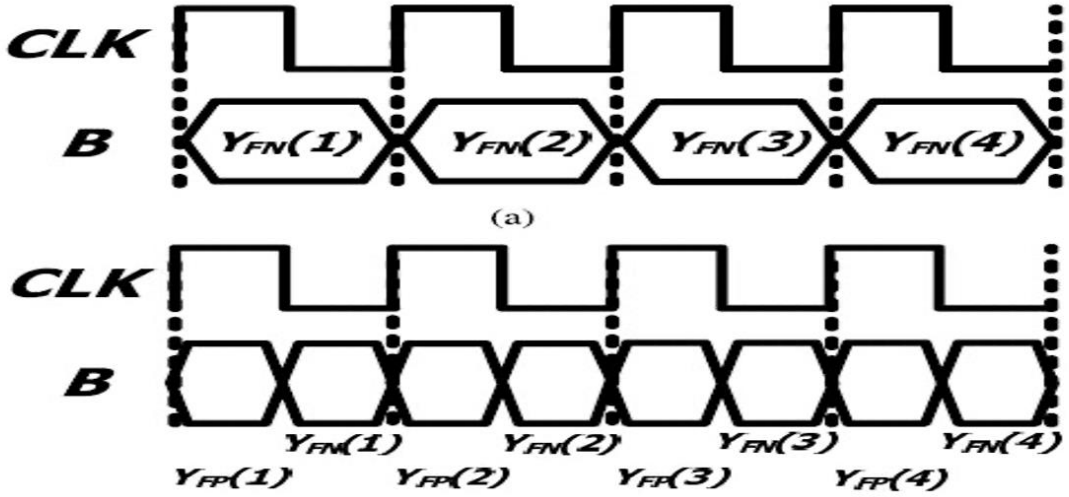

Fig.13. Timing diagrams of feedback path. (a) Feedback path from B to D. (b) Relocated feedback path from C to D

As from the FMO encoder of nm-optimized mode, both YFN(t) and YFP(t) has half latency in it. To solve this another problem another DFF is also applied for YFP(t) as shown in YFP(t). So, condition 1 is satisfied. Both data paths from both DFF's to the multiplexer have no storage component, and this technique has a flexible design space on these data paths which are not symmetrical. Since, the feedback path exists only on YFN(t). So, the feedback path is given to the YFP(t) also as shown in fig.12 As in fig.13 (a), the B is update for positive edge of the clock but in fig.13 (b) it is update both positive and negative edge of the clk. So, now the design is summarized to fig. 14.

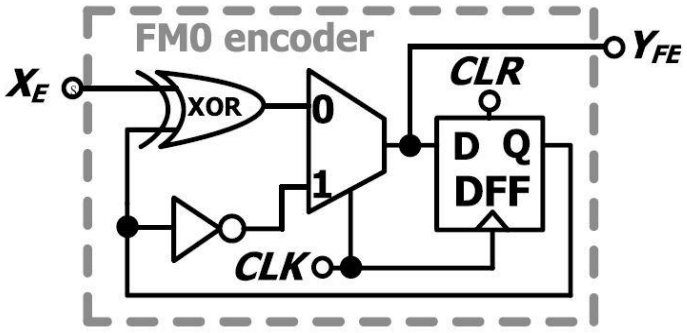

Fig.14. FMO encoder with RBR Technique

For Manchester codec, DFF is directly placed after the multiplexer as in fig 15 YME and XD are on the outputs of the multiplexer and DFF respectively. XD can also be obtained from the multiplexer but used DFF just for compatibility of the design. All the three D flipflops in Fig 10, 14, 15 have 3 common points in the following.

1. They are located after the multiplexer
2. The output of FMO/Manchester encoding, YEE, YME are on the input of a DFF.
3. The outputs of FMO/Manchester decoding, XD are on the output at a DFF.

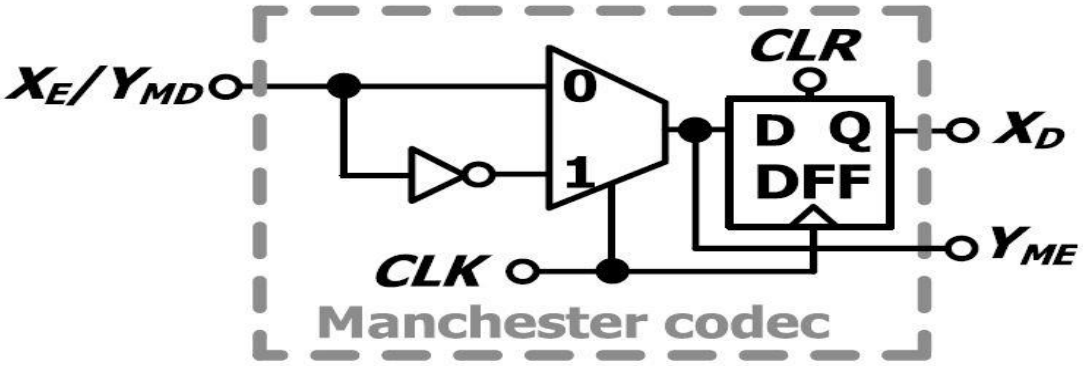

Fig.15. Manchester Codec with RBR Technique

With above three common points, these three DFFs are replaced by one, called the reused DFF, in HCPM. The input of this reused DFF represents YFE/YME, and its output stands for XD, as previously shown in Fig.3.7. So, a single DFF can be used in the final design instead of 3 individual DFF.

3.3. Boolean Function Reshaping (BFR) Technique:

The purpose of BFR is to simplify both positive–cycle logic and integrated negative–cycle logic. These are individually discussed in the following subsections

A. SIMPLIFIED POSITIVE–CYCLE LOGIC:

The negative-cycle logic can be stored into the reused DFF as XD, the positive–cycle logic of FMO decoding YFP is stored into the latch. Moreover, there is no data dependency between positive-cycle logic and negative –cycle logic in FMO/Manchester decoding. Hence, positive –cycle logic of FMO/Manchester decoding can be omitted. As XD is exactly to YMD at negative cycle of CLK. So, it can be omitted.

B. INTEGRATED NEGATIVE-CYCLE LOGIC:

The negative-cycle, XOR in FMO encoding and XNOR in FMO decoding are complex functions. In negative-cycle logic, if the other negative cycle is integrated with them, the HUR is improved greatly. To reach this, other negative-cycle logic is transformed into XNOR function. Reasons for taking XNOR instead of XOR are

1. All negative –cycle logic can be transformed to an uniform function of XNOR with an inversion.
2. This inversion can be reused with the simplified positive-cycle logic, having inversion in FMO/Manchester encoding.

Although originally having a XNOR function, to fit into a XNOR with an inversion, the negative-cycle logic of FMO decoding is further rearranged as

$$\begin{aligned} YFN(t) \_ YFP(t) &= YFN(t) \oplus YFP(t) \\ &= YFN(t)\, YFP(t) + YFN(t)YFP(t) \\ &= YFN(t) \_ YFP(t) \text{----------(3)} \end{aligned}$$

Then, the uniform function of the integrated negative-cycle logic is represented by the following general form

$$m\_\ n,\ m \in M \text{ and } n \in N \text{----------(4)}$$

where m and n individually belong to set M and set N, defined as

$$M = \{XE, YFN(t), 0, YMD\} \text{----------(5)}$$

$$N = \{YFN(t-1), YFP(t), XE, 0\} \text{----------(6)}$$

A general form of the integrated negative-cycle logic is conducted.

With above three core techniques, a brief hardware architecture of FMO/Manchester codec is illustrated in Fig.3.16. The MUXA sequentially presents the simplified positive-cycle logic and the integrated negative-

cycle logic by CLK. There used DFF is conducted by RBR. The inversions of both simplified positive-cycle logic and the integrated negative-cycle logic are reused by the INVA. The MUXB switches between YFN(t−1) and XE for the simplified positive-cycle logic, and SP indicates which one is adopted. The integrated negative-cycle logic is conducted by a XNOR gate, where the data path designs of m and n are discussed as follows. The hardware architecture of the integrated negative-cycle logic is shown in Fig.17. In (5), the M has four elements, where the element '0', logic-0, can be directly derived from there used DFF by enabling CLR, a clear signal, of the DFF. The other three elements in M, XE,YMD, and YFN(t), are from the input of FMO/Manchester codec in Fig.16,whereYFD = {YFP(t), YFN(t)}. The data path of m is realized by MUXC, conducting XE,YMD and YFD in one path and '0' in another. The SN determines which one is passed through MUXC.

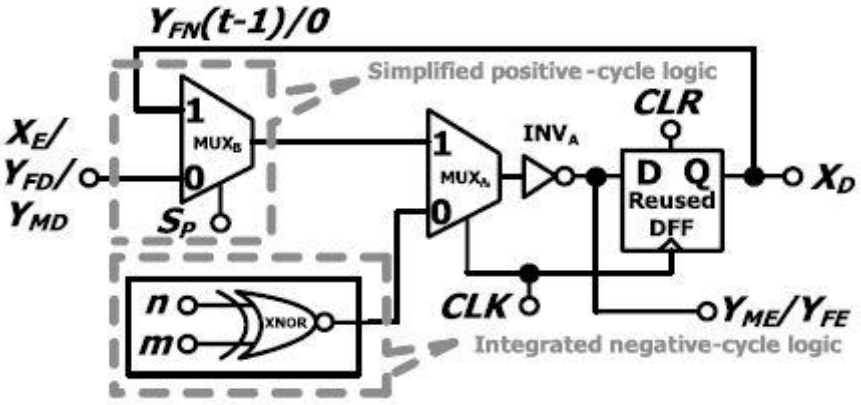

Fig.16. A Brief Architecture of proposed FMO/Manchester Codec

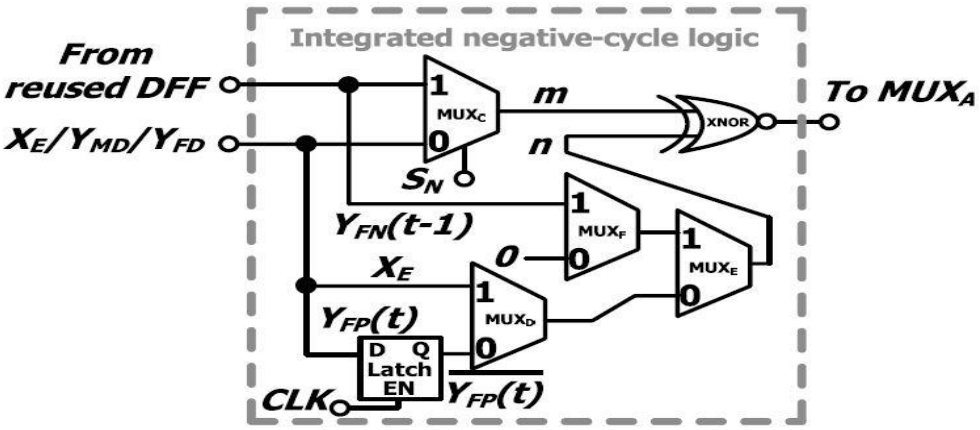

Fig.17. Hardware Architecture of integrated negative cycle

In (6), the N also has four elements of YFN(t−1), YFP(t), XE and '0'. The YFN(t−1) is stored into the reused DFF. The '0' in Nis dedicated for FMO decoding, where the reused DFF is in the charge of sampling the XD. As a result, this '0' cannot be derived from the reused DFF by enabling CLR. An individual 2-to-1 multiplexer is allocated to arrange both YFN(t−1) and the '0'. The XE is from the input of FMO/Manchester codec. As mentioned in Section IV-B, for FMO decoding, the YFP(t) is already stored into the latch at the negative-cycle of CLK. Then, the YFP(t)is obtained from the inverse-output of the latch Q. The data path of n is organized by MUXD, MUXE, and MUXF. Although realizing the integrated negative-cycle logic, this hardware architecture exposes three drawbacks as

1. If YFN(t−1) or '0' is adopted for n, both MUXD and the latch are not in use, which causes a degradation on HUR.
2. MUXD in series of MUXE /MUXF causes the Limitation on processing speed.
3. Three extra control signals for MUXD, MUXE, and MUXF are required.

To overcome above three drawbacks, all four elements in N are reshaped as

$$YFN(t-1) = 0 \cdot YFN(t-1) + 1 \cdot YFN(t-1) \text{ ----------(7)}$$

$$YFP(t) = 1 \cdot YFP(t) + 0 \cdot YFP(t) \text{----------(8)}$$

$$XE = 0 \cdot XE + 1 \cdot XE \text{----------(9)}$$

$$0 = 1 \cdot YMD + 1 \cdot YMD \text{----------(10)}$$

These 4 reshaped Boolean functions can be simply represented by a general form of Out, where Out is a function of 2-to-1 multiplexer listed in Table IV. Both I1 and I0 are two inputs, and the 'Sel' determines which one is passed to the Out. The 'Sel' incorporates YFN(t−1), YFP(t), XE, and YMD. The YFN(t−1) is from the reused DFF, and the others are from the input of FMO/Manchester codec. The data path of 'Sel' is identical to the output of MUXB in Fig.16. Hence, the 'Sel' is directly connected to the output of MUXB without any extra logic. This indicates that the simplified positive-cycle logic is reused with the integrated negative-cycle logic by the output of MUXB. The hardware architecture of integrated negative-cycle logic with reshaped Boolean functions is shown in Fig.18. The Out is realized by MUXD, and the Out is derived from the Q of the latch. Among four elements in N only YFP(t)is required to be stored into the latch, and the other three are purely for the inversion from Q. Also, the placing of the latch after the multiplexer has a consideration of driving capability. More discussion on this part is given in next subsection. This hardware architecture has three advantages listed as

1. Every logic component is fully reused, no matter which element in Nis adopted.
2. Only one multiplexer MUXD is required in the data path of n.
3. The control signal of MUXD is reused with the output of MUXB without any extra logic.

I1	I0	Sel	Out=~ (I1*Sel +$\overline{\iota 0 * Sel}$)
0	1	YFN(t-1)	YFN(t–1)
1	0	YFP(t)	YFP(t)
0	1	XE	XE
1	1	YMD	0

Table.1. 2-To-1 Multiplexer Function in Reshaped Boolean Functions

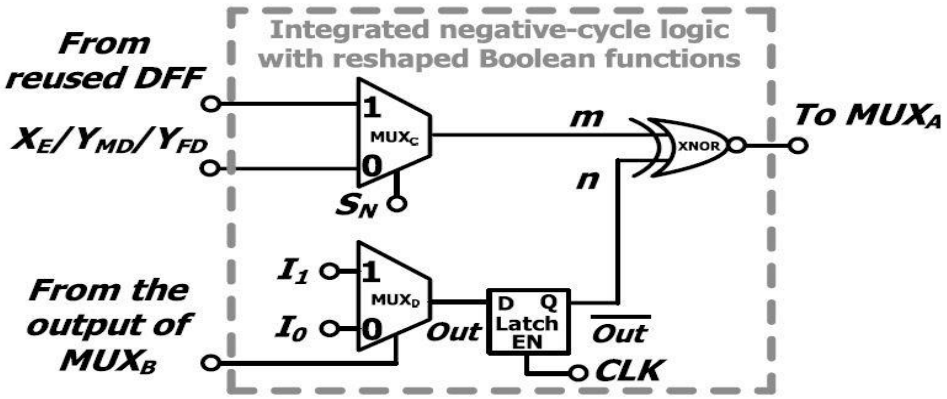

Fig.18. Hardware architecture of integrated negative-cycle logic with reshaped Boolean functions

Coding Mode	SP	SN	I1	I0
FMO Encoding	1	0	1	0
FMO Decoding	0	0	1	1
Manchester Encoding	0	1	1	0
Manchester Decoding	1	0	0	0

Table.3.5. Coding modes of proposed Architecture

The hardware architecture in Fig.18 is integrated into the proposed VLSI architecture of FMO/Manchester codec, as illustrated in Fig.19. The proposed FMO/Manchester codec supports four coding modes, including FMO encoding/decoding and Manchester encoding/decoding. Each coding mode is specified by four parameters SP, SN, I1and I0 as listed in Table V. The output of INVA denotes YFE for FMO encoding or YME for Manchester encoding. Both are valid every half-cycle. The output of reused DFF represents XD for FMO decoding or Manchester decoding and is valid every cycle. Every logic component is fully reused, no matter which coding mode is activated. No logic component is wasted; therefore, the HUR of FMO/Manchester codec is as high as 100%.

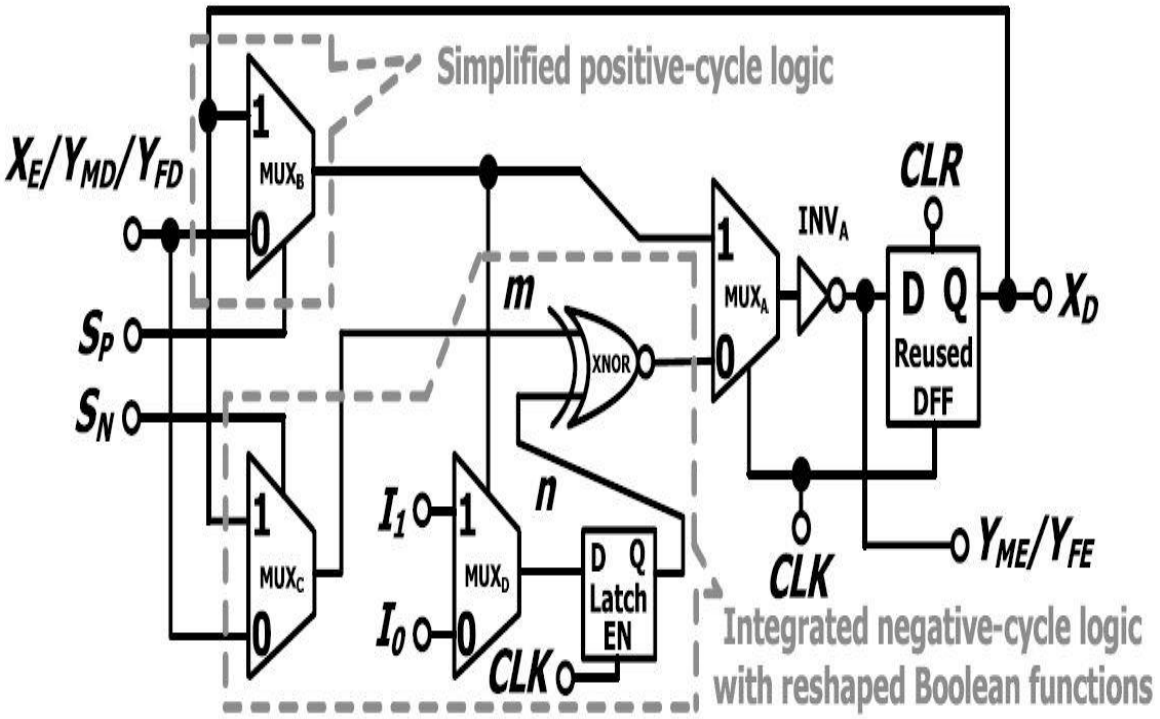

Fig.19. Proposed VLSI Architecture for FMO/Manchester Codec

4. SOFTWARE USED

4.1. Software Description

XILINX ISE (**I**ntegrated **S**ynthesis **E**nvironment) is a software tool produced by Xilinx for synthesis and analysis of HDL designs, enabling the developer to synthesize their designs, perform timing analysis, examine RTL diagrams, simulate a design's reaction to different stimuli, and configure the target device with the programmer.

4.2.User Interface:

The primary user interface of the ISE is the Project Navigator, which includes the design hierarchy (Sources), a Source code editor (Workplace), an output console (Transcript), and a processes tree (Processes). The Design

hierarchy consists of design files (modules), whose dependencies are interpreted by the ISE and displayed as tree structure. For single-chip designs there may be one main module, with other modules included by the main module, like the main() subroutine in C++ programs. Design constraints are specified in modules, which include pin configuration and mapping.

4.3. Simulation:

System-level testing may be performed with ISIM or the Modalism logic simulator, and such test programs must also be written in HDL languages. Test bench programs may include simulated input signal waveforms or monitors which observe and verify the outputs of the device under test. Modalism or ISIM may be used to perform the following types of simulation:

- Logical verification, to ensure the module produces expected results
- Behavioral verification, to verify logical and timing issues
- Post-place & route simulation, to verify behavior after placement of the module within the reconfigurable logic of the FPGA

4.4. Synthesis:

Xilinx's patented algorithms for synthesis allow designs to run up to 30% faster than competing programs and allows greater logic density which reduces project time and costs. Also, due to the increasing complexity of FPGA fabric, including memory blocks and I/O blocks, more complex synthesis algorithms were developed that separate unrelated modules into slices, reducing post-placement errors.

IP Cores are offered by Xilinx and other third-party vendors, to implement system-level functions such as digital signal processing (DSP), bus interfaces, networking protocols, image processing, embedded processors, and peripherals. Xilinx has been instrumental in shifting designs from ASIC-based implementation to FPGA-based implementation.

4.5. Operating Systems:

Xilinx only supports the following operating systems on x86 and x86-64 processor architectures.

- **Microsoft Windows Support**
  - Windows XP Professional (32-bit and 64-bit)
  - Windows 7 Professional (32-bit and 64-bit)
  - Windows Server 2008 (64-bit)
- **Linux Support**
  - Red Hat Enterprise Workstation 5 (32-bit and 64-bit)
  - Red Hat Enterprise Workstation 6 (32-bit and 64-bit)
  - SUSE Linux Enterprise 11 (32-bit and 64-bit)

5. RESULTS

The FMO/Manchester model (Existing model) and the HCPM model (proposed model) has been designed using Verilog HDL coding and simulated using Xilinx 14.3 ISE tool. The Hardware utilization of the two models were analyzed. Different Modes are adopted in CODEC Structure for FMO and MANCHESTER Encoding and Decoding, based on the combinations of SP, SN, I0 and I1 where SP and SN are Select lines to Multiplexer-1 and Multiplexer-2 respectively and I0, I1 are inputs to Multiplexer-3.

5.1 FMO Encoding: The Mode given for FMO encoding is1010 and it can be observed that when input in is 0, a transition is exhibited for every half cycle and when the input is 1, no transition would be exhibited, and it is observed in outd.

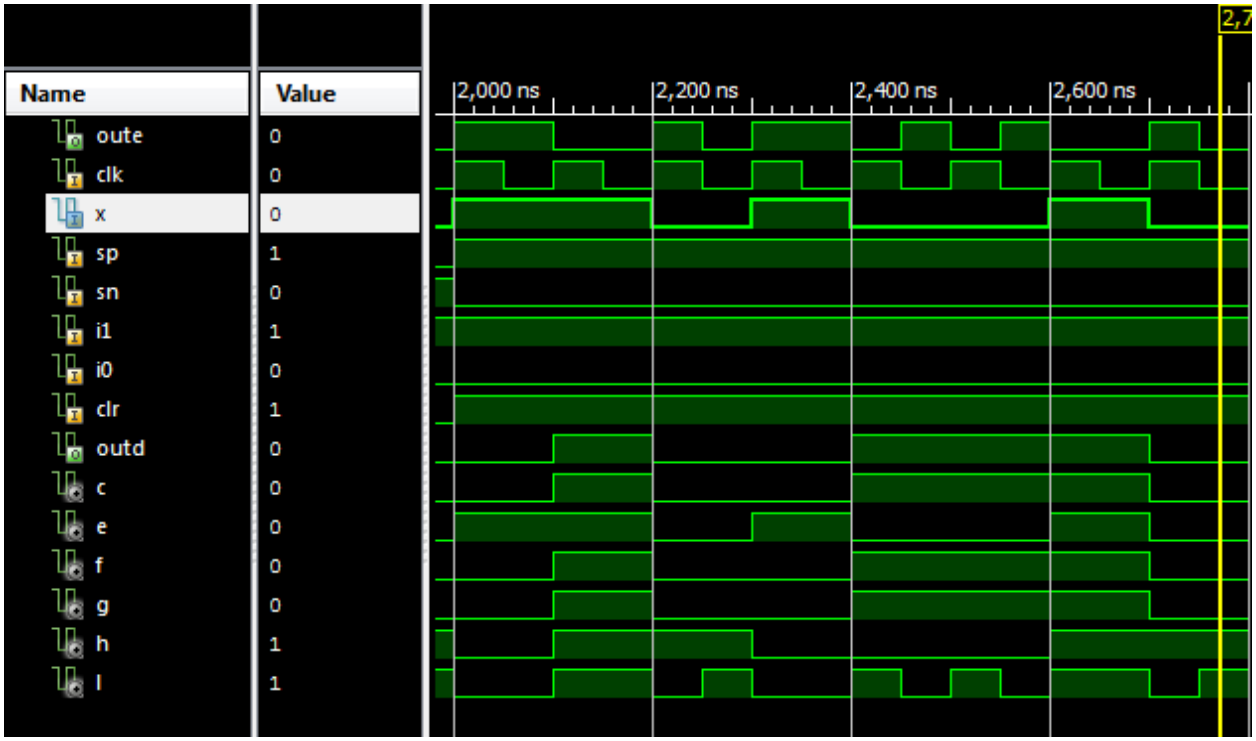

Fig.20. Simulation results of FMO Encoding

5.2 FMO Decoding : The Mode given for FMO decoding is0011 and it can be observed that when input in is 0, the output is 1 if a transition is exhibited for every half cycle at the input and the output is 0 when no transition is exhibited at every cycle of input, and it is observed in outd.

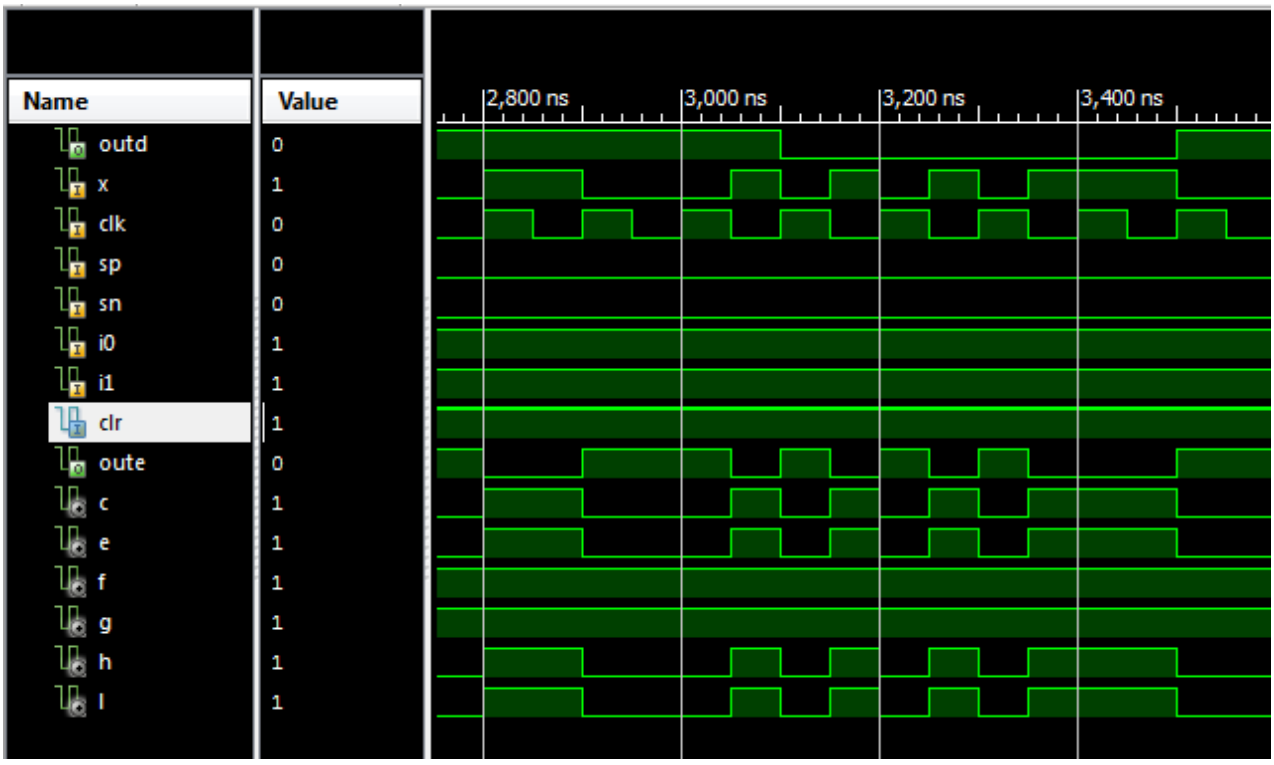

Fig.21. Simulation results of FMO Decoding

5.3 Manchester Encoding : The mode given for Manchester encoding is 0110 and it can be observed that output is obtained by performing XOR operation between clk and input and the output is observed at oute.

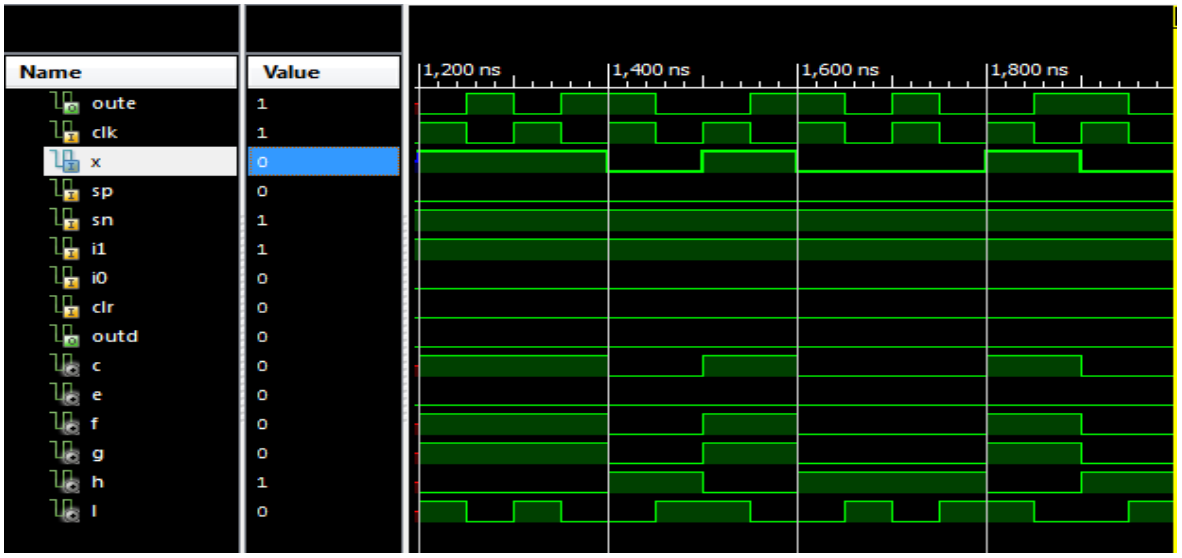

Fig.22. Simulation results of Manchester Encoding

5.4 Manchester Decoding: The mode given for Manchester encoding is 0110 and it can be observed that output is obtained by performing XOR operation between clk and input and the output is observed at outd.

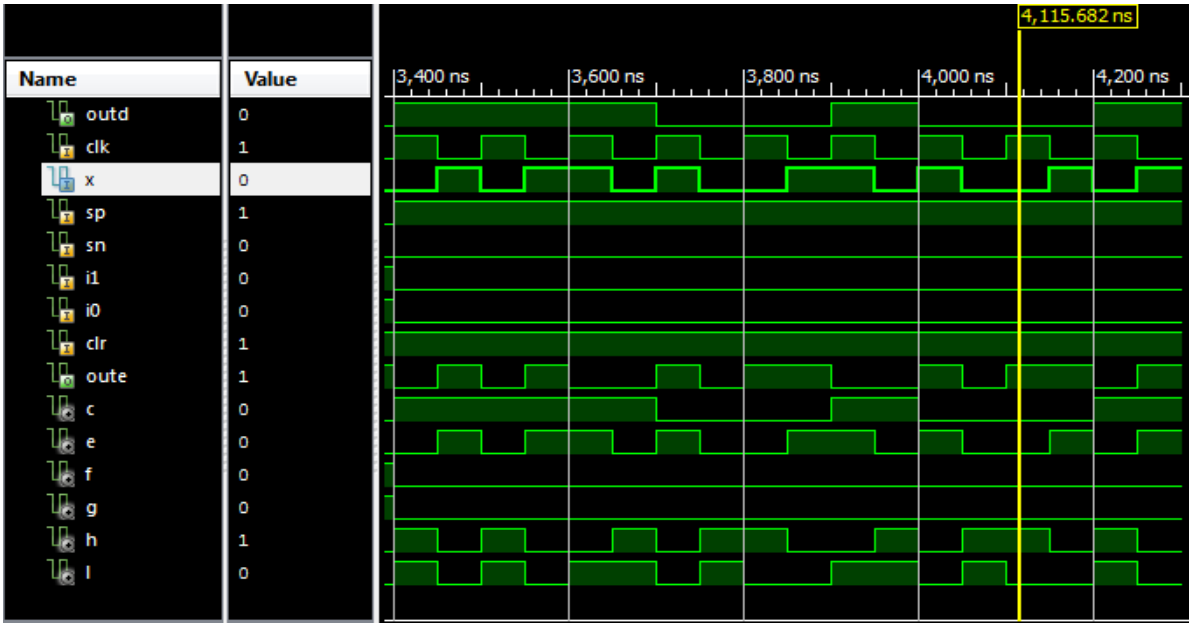

Fig.23. Simulation results of Manchester Decoding

5.5. RTL Schematic:

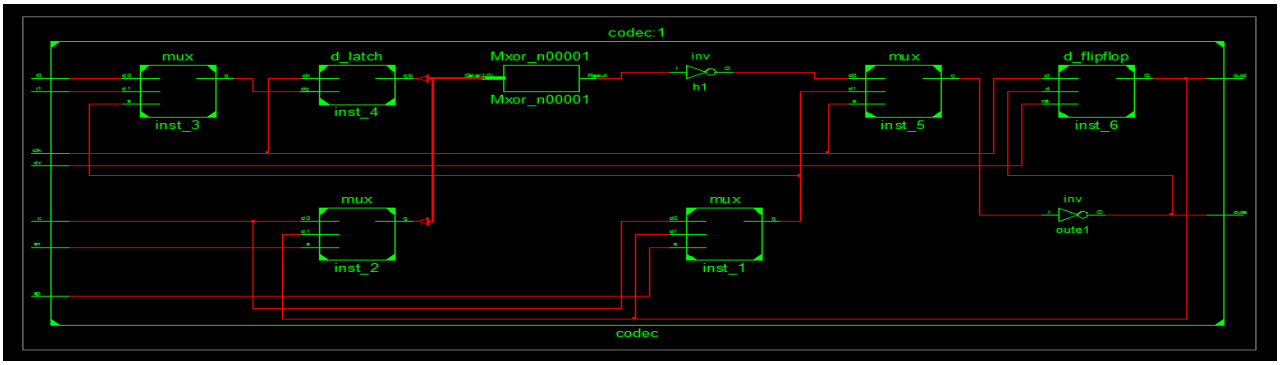

Fig.24. RTL Schematic of Proposed Codec Model

5.6. Hardware Utilization Ratio for Proposed Codec:

Number of Active Transistors: 66
Total Number of Transistors: 66
Where,
D-Flipflop utilizes: 22 transistors.
Mux utilizes: 6 transistors.
XNOR utilizes: 8 transistors.
NOT utilizes: 2 transistors.
D-Latch utilizes: 10 transistors.
HUR= (Number of Active Transistors/Total Number of transistors) *100.
= (66/66) *100
=100 %

5.7. Device Utilization Summary:

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	3	126800	0%
Number of Slice LUTs	6	63400	0%
Number of fully used LUT-FF pairs	0	9	0%
Number of bonded IOBs	10	210	4%
Number of BUFG/BUFGCTRLs	1	32	3%

Fig.25. Utilization Summary of Non optimized Codec

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	2	126800	0%
Number of Slice LUTs	3	63400	0%
Number of fully used LUT-FF pairs	0	5	0%
Number of bonded IOBs	9	210	4%
Number of BUFG/BUFGCTRLs	1	32	3%

Fig.26. Utilization Summary for Proposed Codec Model

6. COMPARISONS:

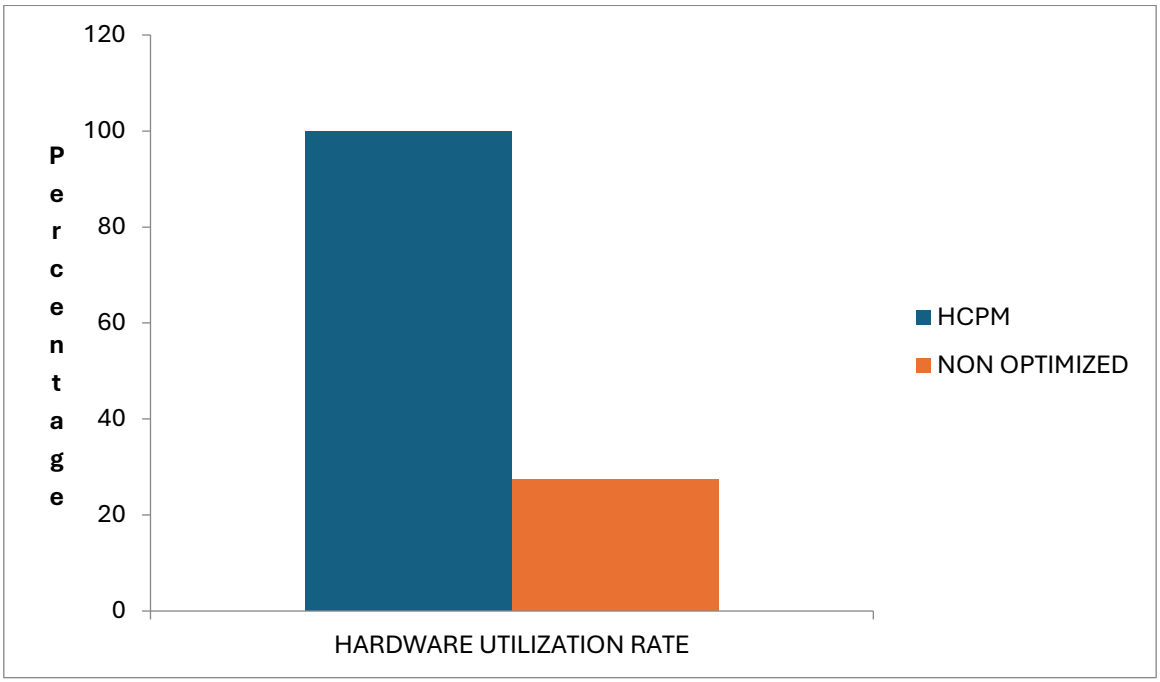

Fig.27. Comparison of HUR of Non-optimized codec and HCPM codec

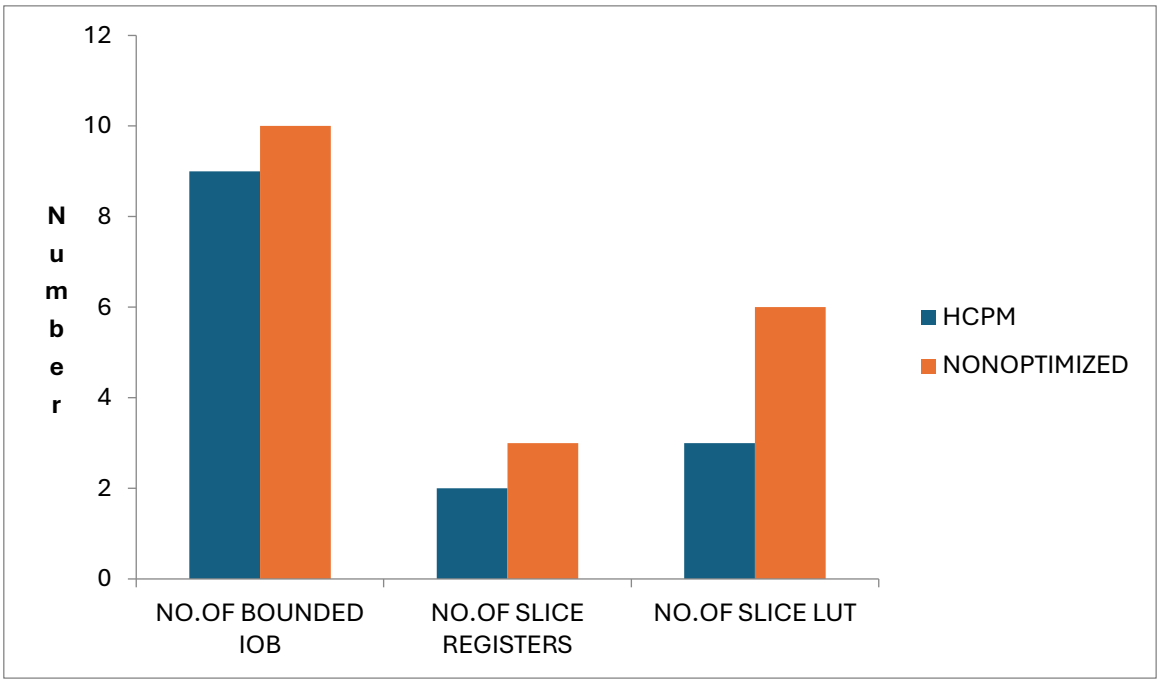

Fig.28. Comparison of Non-optimized codec and HCPM codec

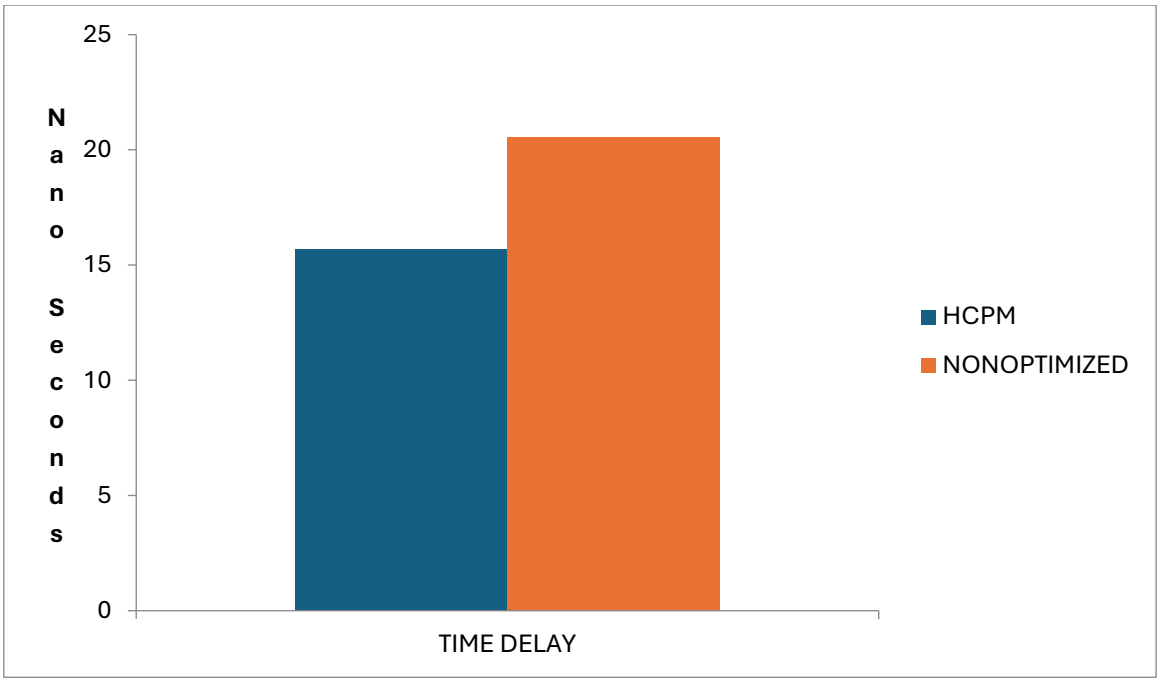

Fig.29. Comparison of Time delay Non-optimized and HCPM codec

7. FUTURE WORK:

The proposed model utilizes four coding techniques namely FMO Encoding/Decoding and Manchester Encoding/Decoding. The future work includes the enhancement of the design so that it can also code another coding technique namely Miller Encoding/Decoding. The reason for choosing Miller coding is that The HCPM in an existing system introduce half cycle latency which can be avoided by using miller codec.

The Miller code itself carries 'timing' information, which can be extracted the clock information in other site, that is, the code with a self-timing property. Therefore, the Miller code has better performance against noise interference. The efficiently realization of the Miller code is thus important research. In open literature, there are not many research for design of CMOS circuit in Miller Coder. The hardware architecture of FMO/Manchester/Miller Codec is shown in Fig.30.

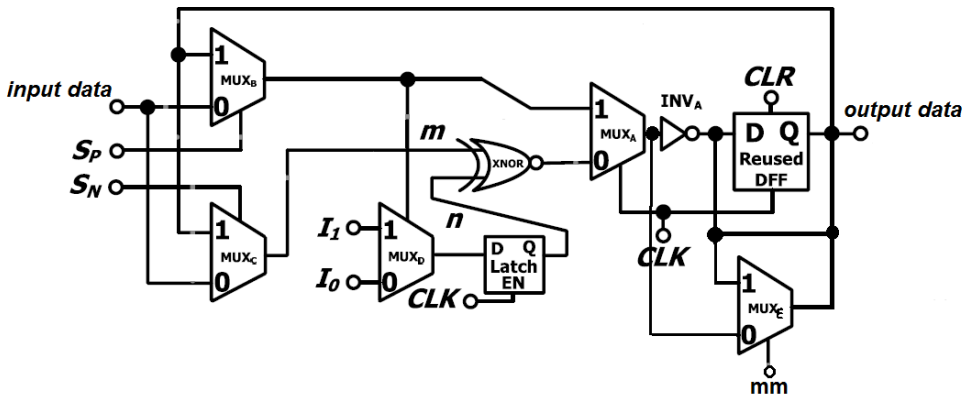

Fig.30. Hardware Architecture of FMO/Manchester/Miller Codec

In [9]SwapnilP.Ramekar and P.N.Chatur proposed "Addition of Miller and Inverted Manchester Encoding Technique to Dedicated Short Range Communication with Full Hardware Utilization", In which hardware architecture for FMO/Manchester/Miller encoding schemes is proposed we are trying to enhance the proposed model to perform both encoding and decoding for all three coding techniques. We achieved Encodings but we

are sorting out methods to perform decoding also which will improve the scope of coding methods used in DSRC.

8. CONCLUSION:

DSRC can provide a wireless link for sensor networking in ITS applications. How to achieve a higher hardware efficiency is a critical consideration to design each essential building-block in sensor node. In this paper, a fully reused VLSI architecture of FMO/Manchester codec with the HUR of 100% is proposed for DSRC-based sensor node. It is based on the HCPM, incorporating HCLP, RBR, and BFR. The HCLP is responsible to classify FMO/Manchester Codec into the positive-cycle logic and negative-cycle logic. The RBR can conduct a reused flip-flop from positive-cycle logic and negative-cycle logic. The BFR further simplifies positive-cycle logic and integrates negative-cycle logic. With the HCPM, the HUR of FMO/Manchester codec is improved from 27.33% to 100%. The experiment results demonstrate that this work presents a competitive performance with100% HUR compared with existing works. With this work, DSRC-based sensor nodes can present a 100% HURFMO/Manchester codec, fully supporting DSRC standards of Europe, USA, and Japan.

9. REFERENCES:

1. P.Benabes, A.Gauthier and J.Oksman, “A Manchester code generator running at 1 GHz,” in Proc. IEEE Int. Conf. Electron., Circuits, Syst., vol. 3. Dec. 2003, pp. 1156–1159.
2. A.Karagounis, A.Polyzos, B.Kotsos, and N.Assimakis, “A 90 nm Manchester code generator with CMOS switches running at 2.4 GHz and 5 GHz,” in Proc. 16th Int. Conf. Syst., Signals, Image Process., Jun. 2009, pp. 1–4.
3. W.M.El.Medany, “FPGA implementation of RDR Manchester and D-Manchester CODEC design for wireless transceiver,” in Proc. Nat. Radio Sci. Conf., Mar. 2008, pp. 1–5.
4. Y.C.Hung, M.M.Kuo, C.K.Tung, and S.H.Shieh, “High-speed CMOS chip design for Manchester and Miller encoder,” in Proc. 5th Int. Intell. Inf. Hiding Multimedia Signal Process., Sep. 2009, pp. 538–541.
5. M.A.Khan, M.Sharma, and P.R.Brahmanandha, “FSM based Manchester encoder for UHF RFID tag emulator,” in Proc. Int. Conf. Comput., Commun., Netw., Dec. 2008, pp. 1–6.
6. J.H.Deng, F.C.Hsiao and Y.H.Lin, “Top down design of joint MODEM and CODEC detection schemes for DSRC coded-FSK systems over high mobility fading channels,” in Proc. 15th Int. Conf. Adv.Commun. Technol., Jan. 2013, pp. 98–103.
7. F.N.Zhang, X.A.Wang, S.S.Yong, X.L.Shi, B.Liu and Z.Y.Guo, “A multi-bit encoder and FMO/Miller decoder design for UHF RFID reader digital baseband,” in Proc. IEEE 11th Int. Conf. Solid-State Integr. Circuit Technol., Oct./Nov. 2012, pp. 1–3.
8. N.F.B.Bautista and J.J.S.Marciano Jr., “Enhanced FMO decoder for UHF passive RFID readers using duty cycle estimations,” in Proc. IEEE Int. Conf. RFID-Technol. Appl., Sep. 2011, pp. 306–312.
9. Swapnil P.Ramekar and P.N.Chatur, “Addition of Miller and Inverted Manchester Encoding Technique to Dedicated Short Range Communication with Full Hardware Utilization”, in 2016 International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), Jan.2016.