Fully Reused VLSI Architecture of FM0/Manchester Encoding for DSRC Applications by Using SOLS Technique

P. Krishnasri¹, D. Gopinath², Rajaiah Gabbeta³

¹M.Tech Student, Abdulkalam Institute of Technological Sciences, Kothagudem, Telangana, India.
²Asst Prof, Dept of ECE, Abdulkalam Institute of Technological Sciences, Kothagudem, Telangana, India.
³Professor & Head of Department of ECE, Abdulkalam Institute of Technological Sciences, Kothagudem, Telangana, India.

Abstract—The dedicated short-range communication (DSRC) is an emerging technique to push the intelligent transportation system into our daily life. The DSRC standards generally adopt FM0 and Manchester codes to reach dc-balance, enhancing the signal reliability. Nevertheless, the coding-diversity between the FM0 and Manchester codes seriously limits the potential to design a fully reused VLSI architecture for both. In this paper, the similarity-oriented logic simplification (SOLS) technique is proposed to overcome this limitation. The SOLS technique improves the hardware utilization rate from 57.14% to 100% for both FM0 and Manchester encodings. The performance of this paper is evaluated on the postlayout simulation in Taiwan Semiconductor Manufacturing Company (TSMC) 0.18-µm 1P6M CMOS technology. The maximum operation frequency is 2 GHz and 900 MHz for Manchester and FM0 encodings, respectively. The power consumption is 1.58 mW at 2 GHz for Manchester encoding and 1.14 mW at 900 MHz for FM0 encoding. The core circuit area is 65.98 × 30.43 m². The encoding capability of this paper can fully support the DSRC standards of America, Europe, and Japan. This paper not only develops a fully reused VLSI architecture, but also exhibits an efficient performance compared with the existing works.

Index Terms — Dedicated short-range communication (DSRC), FM0, Manchester, VLSI.

I. INTRODUCTION

The dedicated short-range communication (DSRC) [1] is a protocol for one- or two-way medium range communication especially for intelligent transportation systems. The DSRC can be briefly classified into two categories: automobile-to-automobile and automobile-to-roadside. In automobile-to-automobile, the DSRC enables the message sending and broadcasting among automobiles for safety issues and public information announcement [2], [3]. The safety issues include blind-spot, intersection warning, intercars distance, and collision-alarm. The automobile-to-roadside focuses on the intelligent transportation service, such as electronic toll collection (ETC) system. With ETC, the toll collecting is electrically accomplished with the contactless IC-card platform. Moreover, the ETC can be extended to the payment for parking-service, and gas-refueling. Thus, the DSRC system plays an important role in modern automobile industry. The system architecture of DSRC transceiver is shown in Fig. 1. The upper and bottom parts are dedicated for transmission and receiving, respectively. This transceiver is classified into three basic modules: microprocessor, baseband processing, and RF frontend. The microprocessor interprets instructions from media access control to schedule the tasks of baseband processing and RF front-end. The baseband processing is responsible for modulation, error correction, clock synchronization, and encoding. The RF frontend transmits and receives the wireless signal through the antenna.

Fig 1: System architecture of DSRC transceiver.

The DSRC standards have been established by several organizations in different countries. These DSRC standards of America, Europe, and Japan are shown in Table I. The data rate individually targets at 500 kb/s, 4 Mb/s, and 27 Mb/s with carrier frequency of 5.8 and 5.9 GHz. The modulation methods incorporate amplitude shift keying, phase shift keying, and orthogonal frequency division multiplexing. Generally, the waveform of transmitted signal is expected to have zero mean for robustness issue, and this is also referred to as dc-balance. The transmitted signal consists of arbitrary binary sequence, which is difficult to obtain dc-balance. The purposes of FM0 and Manchester codes can provide the transmitted signal with dc-balance.

Both FM0 and Manchester codes are widely adopted in encoding for downlink. The VLSI architectures of FM0 and Manchester encoders are reviewed as follows.

A. Review of VLSI Architectures for FM0 Encoder and Manchester Encoder

The literature [4] proposes a VLSI architecture of Manchester encoder for optical communications. This design adopts the CMOS inverter and the gated inverter as the switch to construct Manchester encoder. It is implemented by 0.35-μm CMOS technology and its operation frequency is 1 GHz. The literature [5] further replaces the architecture of switch in [4] by the nMOS device. It is realized in 90-nm CMOS technology, and the maximum operation frequency is as high as 5 GHz. The literature [6] develops a high-speed VLSI architecture almost fully reused with Manchester and Miller encodings for radio frequency identification (RFID) applications. This design is realized in 0.35-μm CMOS technology and the maximum operation frequency is 200 MHz. The literature [7] also proposes a Manchester encoding architecture for ultrahigh frequency (UHF) RFID tag emulator. This hardware architecture is conducted from the finite state machine (FSM) of Manchester code, and is realized into field-programmable gate array (FPGA) prototyping system. The maximum operation frequency of this design is about 256 MHz. The similar design methodology is further applied to individually construct FM0 and Miller encoders also for UHF RFID Tag emulator [8]. Its maximum operation frequency is about 192 MHz. Furthermore, [9] combines frequency shift keying (FSK) modulation and demodulation with Manchester code in hardware realization.

B. Features of This Paper

However, the coding-diversity between both seriously limits the potential to design a VLSI architecture that can be fully reused with each other. This paper proposes a VLSI architecture design using similarity-oriented logic simplification (SOLS) technique. The SOLS consists of two core methods: area-compact retiming and balance logic-operation sharing. The area-compact retiming relocates the hardware resource to reduce 22 transistors.

![Fig 2: Codeword structure of FM0](image)

The balance logic-operation sharing efficiently combines FM0 and Manchester encodings with the fully reused hardware architecture. With SOLS technique, this paper constructs a fully reused VLSI architecture of Manchester and FM0 encodings for DSRC applications. The experiment results reveal that this design achieves an efficient performance compared with sophisticated works.

C. Organization

The remainder of this paper is organized as follows. Section II describes the coding principles of FM0 and Manchester codes. Section III gives a limitation analysis on hardware utilization of FM0 and Manchester encoders. This section shows the difficulty to design a fully reused VLSI architecture for FM0 and Manchester encoders. The proposed VLSI architecture design using SOLS technique is reported in Section IV. Two core methods of SOLS technique, area compact retiming and balance logic-operation sharing, are described in this section. The experiment results and discussion are presented in Section V. This section focuses on an objective evaluation between this design and existing articles of Manchester and FM0 encoders. Finally, the conclusion is given in Section VI.

II. CODING PRINCIPLES OF FM0 CODE AND MANCHESTER CODE

In the following discussion, the clock signal and the input data are abbreviated as CLK, and X, respectively. With the above parameters, the coding principles of FM0 and Manchester codes are discussed as follows.

A. FM0 Encoding

As shown in Fig. 2, for each X, the FM0 code consists of two parts: one for former-half cycle of CLK, A, and the other one for later-half cycle of
CLK, B. The coding principle of FM0 is listed as the following three rules.
1) If X is the logic-0, the FM0 code must exhibit a transition between A and B.
2) If X is the logic-1, no transition is allowed between A and B.
3) The transition is allocated among each FM0 code no matter what the X is. A FM0 coding example is shown in Fig. 3. At cycle 1, the X is logic-0; therefore, a transition occurs on its FM0 code, according to rule 1. For simplicity, this transition is initially set from logic-0 to -1. According to rule 3, a transition is allocated among each FM0 code, and

![Fig 3: Illustration of FM0 coding example.](image)

Thus, the logic-1 is changed to logic-0 in the beginning of cycle 2. Then, according to rule 2, this logic-level is hold without any transition in entire cycle 2 for the X of logic-1. Thus, the FM0 code of each cycle can be derived with these three rules mentioned earlier.

B. Manchester Encoding

The Manchester coding example is shown in Fig. 4. The Manchester code is derived from

\[ X \oplus CLK \] (1)

The Manchester encoding is realized with a XOR operation for CLK and X. The clock always has a transition within one cycle, and so does the Manchester code no matter what the X is.

![Fig 4: Illustration of Manchester coding example.](image)

III. LIMITATION ANALYSIS ON HARDWARE UTILIZATION OF FM0 ENCODER AND MANCHESTER ENCODER

To make an analysis on hardware utilization of FM0 and Manchester encoders, the hardware architectures of both are conducted first. As mentioned earlier, the hardware architecture of Manchester encoding is as simple as a XOR operation. However, the conduction of hardware architecture for FM0 is not as simple as that of Manchester. How to construct the hardware architecture of FM0 encoding should start with the FSM of FM0 first. As shown in Fig. 5(a), the FSM of FM0 code is classified into four states. A state code is individually assigned to each state, and each state code consists of A and B, as shown in Fig. 2. According to the coding principle of FM0, the FSM of FM0 is shown in Fig. 5(b). Suppose the initial state is \( S_1 \), and its state code is 11 for A and B, respectively. If the X is logic-0, the state-transition must follow both rules 1 and 3. The only one next-state that can satisfy both rules for the X of logic-0 is \( S_3 \).

![Fig 5: Illustration of FSM for FM0. (a) States definition. (b) FSM of FM0.](image)
The FSM of FM0 can also conduct the transition table of each state, as shown in Table II. A(t) and B(t) represent the discrete-time state code of current-state at time instant t. Their previous-states are denoted as the A(t−1) and the B(t−1), respectively. With this transition table, the Boolean functions of A(t) and B(t) are given as

\[
A(t) = B(t-1) (2) \\
B(t) = X \oplus B(t-1) (3)
\]

With both A(t) and B(t), the Boolean function of FM0 code is denoted as

\[
CLK A(t) + \overline{CLK} B(t). \quad (4)
\]

With (1) and (4), the hardware utilization rate (HUR) is defined as

\[
HUR = \frac{Active\ Components}{Total\ Components} \times 100\% \quad (5)
\]

The coding-diversity between the FM0 and Manchester codes seriously limits the potential to design a fully reused VLSI architecture.

IV. VLSI ARCHITECTURE DESIGN OF FM0 ENCODER AND MANCHESTER ENCODER USING SOLS TECHNIQUE

The purpose of SOLS technique is to design a fully reused VLSI architecture for FM0 and Manchester encodings. The SOLS technique is classified into two parts: area-compact retiming and balance logic-operation sharing. Each part is individually described as follows. Finally, the performance evaluation of the SOLS technique is given.

A. Area-Compact Retiming

The FM0 logic in Fig. 6 is simply shown in Fig. 7(a). The logic for A(t) and the logic for B(t) are the Boolean functions to derive A(t) and B(t), where the X is omitted for a concise representation. For FM0, the state code of each state is stored into DFFA and DFFB. The determination of which coding is adopted depends on the Mode selection of the MUX−2, where the Mode = 0 is for FM0 code, and the Mode = 1 is for Manchester code. To evaluate the hardware utilization, the hardware utilization rate (HUR) is defined as

\[
HUR = \frac{Active\ Components}{Total\ Components} \times 100\% \quad (5)
\]
Thus, the FM0 encoding just requires a single 1-bit flip-flop to store the $B(t-1)$. If the DFFA is directly removed, a nonsynchronization between $A(t)$ and $B(t)$ causes the logic fault of FM0 code. To avoid this logic-fault, the DFFB is relocated right after the MUX−1, as shown in Fig. 7(b), where the DFFFB is assumed to be positive-edge triggered. At each cycle, the FM0 code, comprising $A$ and $B$, is derived from the logic of $A(t)$ and the logic of $B(t)$, respectively. The FM0 code is alternatively switched between $A(t)$ and $B(t)$ through the MUX−1 by the control signal of the CLK. In Fig. 7(a), the $Q$ of DFFB is directly updated from the logic of $B(t)$ with 1-cycle latency. In Fig. 7(b), when the CLK is logic-0, the $B(t)$ is passed through MUX−1 to the $D$ of DFFF. Then, the upcoming positive-edge of CLK updates it to the $Q$ of DFFF. As shown in Fig. 8, the timing diagram for the $Q$ of DFFF is consistent whether the DFF is relocated or not. Suppose the logic components of FM0 encoder are realized with the logic-family of static CMOS, and the total transistor count is shown in Table IV. The transistor count of the FM0 encoding architecture without area-compact retiming is 72, and that with area-compact retiming is 50. The area-compact retiming technique reduces 22 transistors.

### B. Balance Logic-Operation Sharing

As mentioned previously, the Manchester encoding can be derived from $X \oplus \text{CLK}$, and it is also equivalent to

$$X \oplus \text{CLK} = X \text{CLK} + \overline{X} \text{CLK}. \quad (6)$$

This can be realized by the multiplexer, as shown in Fig. 9(a). It is quite similar to the Boolean function of FM0 encoding in (4). By comparing with (4) and (6), the FM0 and Manchester logics have a common point of the multiplexer like logic with the selection of CLK. As shown in Fig. 9(b), the concept of balance logic-operation sharing is to integrate the $X$ into $A(t)$ and $X$ into $B(t)$, respectively. The logic for $A(t)/X$ is shown in Fig. 10.

![Fig 8: Timing diagram of area-compact retiming for FM0 encoding](image8)

![Fig 9: Concept of balance logic-operation sharing for FM0 and Manchester encodings. (a) Manchester code. (b) Combines the logic operations of Manchester and FM0 encodings.](image9)

![Fig 10: Balance logic-operation sharing of $A(t)$ and $X$.](image10)
operation can be shared with Manchester and FM0 encodings. As a result, the logic for \( B(t)/X \) is shown in Fig. 11(b), where the multiplexer is responsible to switch the operands of \( B(t-1) \) and logic-0. This architecture shares the XOR for both \( B(t) \) and \( X \), and thereby increases the HUR. Furthermore, the multiplexer in Fig. 11(b) can be functionally integrated into the relocated DFFB from area-compact retiming technique, as shown in Fig. 11(c). The CLR is the clear signal to reset the content of DFFB to logic-0. The DFFB can be set to zero by activating CLR for Manchester encoding. When the FM0 code is adopted, the CLR is disabled, and the \( B(t-1) \) can be derived from DFFB. Hence, the multiplexer in Fig. 11(b) can be totally saved, and its function can be completely integrated into the relocated DFFB. The proposed VLSI architecture of FM0/Manchester encoding using SOLS technique is shown in Fig. 12(a). The logic for \( A(t)/\overline{X} \) includes the MUX-2 and an inverter. Instead, the logic for \( B(t)/X \) just incorporates a XOR gate.

In the logic for \( A(t)/\overline{X} \), the computation time of MUX-2 is almost identical to that of XOR in the logic for \( B(t)/X \). However, the logic for \( A(t)/\overline{X} \) further incorporates an inverter in the series of MUX-2. This unbalance computation time between \( A(t)/\overline{X} \) and \( B(t)/X \) results in the glitch to MUX-1, possibly causing the logic-fault on coding. To alleviate this unbalance computation time, the architecture of the balance computation time between \( A(t)/X \) and \( B(t)/X \) is shown in Fig. 12(b). The XOR in the logic for \( B(t)/X \) is translated into the XNOR with an inverter, and then this inverter is shared with that of the logic for \( A(t)/\overline{X} \). This shared inverter is relocated backward to the output of MUX-1. Thus, the logic computation time between \( A(t)/\overline{X} \) and \( B(t)/X \) is more balance to each other. The adoption of FM0 or Manchester code depends on Mode and CLR. In addition, the CLR further has another individual function of a hardware initialization. If the CLR is simply derived by inverting Mode without assigning an individual CLR control signal, this leads to a conflict between the coding mode selection and the hardware initialization. To avoid this conflict, both Mode and CLR are assumed to be separately allocated to this design from a system controller. Whether FM0 or Manchester code is adopted, no logic component of the proposed VLSI architecture is wasted. Every component is active in both FM0 and Manchester coding.
encodings. Therefore, the HUR of the proposed VLSI architecture is greatly improved.

C. Timing Analysis

The logic functions of SOLS technique can be realized by various logic families. Each logic family optimizes one or more electrical performance, such as area, power, or speed, from circuit topology perspective instead of architecture perspective. The proposed SOLS technique is developed from architecture perspective to achieve 100% HUR. Among the logic families, both static CMOS circuit and transmission gate logic are widely applied in digital circuit owing to their superior integration in process manufacturing. Hence, the timing analysis is given under these two kinds of logic families for a more general purpose. Although the SOLS technique enables the VLSI architecture to be fully shared for 2not identical. For Manchester encoding, the delay time is given as

$$T_{\text{Man}} = \max\{ T_{\text{MUX}}, T_{\text{XNOR}} \} + T_{\text{INV}} + T_{\text{DDFF}} \quad (7)$$

where $T_{\text{Man}}$ denotes the delay time of Manchester encoding. The $T_{\text{MUX}}$, $T_{\text{XNOR}}$, and $T_{\text{DDFF}}$ represent the delay time of the multiplexer, the XNOR gate, and the inverter, respectively. The DFFB is always kept at logic-0 in Manchester encoding; therefore, it is excluded from $T_{\text{Man}}$. This delay path is also incorporated into that of FM0 encoding. Moreover, the FM0 encoding applies the DFFB to store the sate code, and thereby the delay time of DFFB is further considered as

$$T_{\text{FM0}} = T_{\text{Man}} + T_{\text{DDFF}} \quad (8)$$

where the $T_{\text{FM0}}$ is the delay time of FM0 encoding, and the $T_{\text{DDFF}}$ stands for the delay time of DFFB. The delay time of Manchester encoding is smaller than that of FM0 encoding. Thus, the operation frequency of Manchester encoding is faster than that of the FM0 encoding in the proposed VLSI architecture. From the above timing analysis, the $T_{\text{Man}}$ not only dominates the timing of Manchester encoding, but also affects that of FM0 encoding. Hence, the logic components inside $T_{\text{Man}}$ should be carefully considered in their implementation. If these logic components are totally designed with static CMOS, the $T_{\text{Man}}$ is seriously limited owing to too many transistors in the critical path of Manchester encoding. More detail on this part is described as follows.

For simplicity, both rise and fall times of static CMOS circuit are assumed to be identical. The fall time is adopted to denote the propagation delay with Elmore delay estimation. The static CMOS topologies of two-input multiplexer and two input XNOR are shown in Fig. 13(a) and (b), respectively. The pull-down network of two-input multiplexer includes $M_1, M_2, M_3$ and $M_4$. The $M_1$ and $M_2$ are in parallel, and so are $M_3$ and $M_4$. Indeed, this connection can improve the discharging capability. However, the transistor sizing still considers the worst case where the discharging path is constructed by only two transistors in series. The propagation delay of the static CMOS two-input multiplexer is given as

$$T_{\text{MUX-SC}} = T_{\text{INV}} + C_1R + C_22R \quad (9)$$

where the second and third terms are predicted by Elmore delay estimation. The $R$ stands for the equivalent resistor of each transistor in pull-down network. The $C_A$ aggregates the junction capacitance of $M_1, M_2$, and $M_4$. The $C_B$ gathers the junction capacitance of $M_3, M_5, M_3$, and $M_6$. Similarly, the propagation delay of static CMOS two-input XNOR is given as

$$T_{\text{XNOR-SC}} = T_{\text{INV}} + C_1R + C_22R \quad (10)$$

Compared with (9), the $C_A$ is greater than $C_C$ since the $C_A$ incorporates more junction area than $C_C$, and the $C_B$ approximates to $C_D$. Thus, the $T_{\text{MUX-SC}}$ is greater than $T_{\text{XNOR-SC}}$.

Fig 13: Static CMOS topologies of multiplexer and XNOR. (a) Two-input multiplexer. (b) Two-input XNOR.

In Fig. 12(b), for static CMOS, the series of MUX-1 and MUX-2 dominates Manchester encoding path and leads to a total propagation delay as

$$2T_{\text{MUX-SC}} = 2(T_{\text{INV}} + C_1R + C_22R) \quad (11)$$
To further reduce the transistor count in Manchester encoding path, the transmission-gate logic is considered in the circuit designs of MUX−1, MUX−2 and XNOR. The transmission gate logic of two-input multiplexer and two-input XNOR are shown in Fig. 14(a) and (b), respectively.

The propagation delay of transmission-gate two-input multiplexer is given as

\[ T_{\text{MUX-TG}} = R' C_E \]  \hspace{1cm} (12)

where the \( CE \) aggregates the junction capacitances of \( M_1, M_2, M_3 \) and \( M_4 \). The \( R' \) stands for the equivalent resistances of \( M_1 \) and \( M_2 \) in parallel, or \( M_3 \) and \( M_4 \) in parallel. Similarly, the propagation delay of transmission-gate two-input XNOR is given as

\[ T_{\text{XNOR-TG}} = T_{\text{INV}} + R' C_F \]  \hspace{1cm} (13)

where the \( C_F \) gathers the junction capacitances of \( M_1, M_2, M_3 \) and \( M_4 \). The \( T_{\text{XNOR-TG}} \) causes slightly longer propagation delay than \( T_{\text{MUX-TG}} \).

In Fig. 12(b), for transmission-gate logic, the series of XNOR and MUX−2 dominates Manchester encoding path and leads to a total propagation delay as

\[ T_{\text{XNOR-TG}} + T_{\text{MUX-TG}} = T_{\text{INV}} + R' C_E + R' C_F \]  \hspace{1cm} (14)

For a concise comparison with (11), the assumptions are given as follows. The \( C_A, C_B, C_C, C_D \) and \( C_E \) consistently incorporate the junction capacitances of two nMOS and two pMOS. All these four capacitors can be assumed as \( C_B = C_D = C_E = C_F \). For static CMOS circuit, both two-input multiplexer and two-input XNOR have two nMOS in series. The width of these two nMOS in series is enlarged by two times to achieve a balance fall-time, and the \( R_N \) is reduced down to \( R_{N}/2 \). A transmission gate consisting of a pMOS and an nMOS in parallel has its equivalent resistor of \( R_P/2 \). To obtain a balance signal path, \( R_P \) is approximated to \( R_N \) by enlarging the width of pMOS by two to three times depending on the process technology. Hence, the equivalent resistor of a transmission gate is \( R_P/2 \). The equivalent resistor of an nMOS in two-input multiplexer or two-input XNOR is identical to that of a transmission gate. Thus, the assumption is given as \( R = R' = R_{N}/2 \). With above assumptions, the propagation delay of transmission-gate logic is less than that of static CMOS. Applying the transmission-gate logic can compact the transistor count to reduce the propagation delay. Generally, the buffer is inserted every three stage of transmission-gate logic [10], and this rule is also applied in the algorithm of buffer-insertion [11]. In Fig. 12(b), accordingly, the circuit architectures of MUX−1, MUX−2, and XNOR are designed with transmission-gate logic. Hence, only two-stage of transmission-gate logic is adopted, and no extra buffer-insertion is required in this design.

V. SIMULATION RESULTS

The simulations of the designs are carried out by using Verilog HDL language in modelsim software. The simulated waveforms of the proposed designs are shown in below figures.

Fig 14: Transmission-gate multiplexer and XNOR.
(a) Two-input multiplexer. (b) Two-input XNOR

Fig 15: Top cov simulation results

Fig 16: Top1 simulation results
VI. CONCLUSION

The coding-diversity between FM0 and Manchester encodings causes the limitation on hardware utilization of VLSI architecture design. A limitation analysis on hardware utilization of FM0 and Manchester encodings is discussed in detail. In this paper, the fully reused VLSI architecture using SOLS technique for both FM0 and Manchester encodings is proposed. The SOLS technique eliminates the limitation on hardware utilization by two core techniques: areacompact retiming and balance logic-operation sharing. The encoding capability of this paper can fully support the DSRC standards of America, Europe, and Japan. This paper not only develops a fully reused VLSI architecture, but also exhibits a competitive performance compared with the existing works.

REFERENCES


AUTHORS:

P. Krishnasri has received her B.Tech degree in Electronics and communication engineering from Sreeekavitha Engg College, Karepalli, Khammam in 2010 and currently she is pursuing her M.Tech specialization in EMBEDDED SYSTEMS & VLSI system design in Abdulkalam Institute Of Technological Sciences, Vepalagadda, JNTUH.

Rajaiah Gabbeta has received his B.Tech degree in Electronics and Instrumentation from Kakatiya University in 1997 and M.Tech degree in Instrumentation and Control Systems from JNTU Kakinada in 2005. He has been working as Professor& Head of Department. He has contributed more than 20 reviewed publications in journals. Current projects in the area of Image based retrieval, Historical handwriting analysis, and forensic handwriting analysis systems.

D.Gopinath has received his B.Tech degree in Electronics and Communication Engineering from JNTU, Hyderabad in 2010 and M.Tech degree in Electronics and Communication Engineering with specialization of VLSI SYSTEM DESIGN from JNTU Hyderabad in 2013. Presently working as Assistant Professor at Abdulkalam Institute of Technological Sciences, Kothagudem, telangana state.