International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 03
March 2017

VLSI Architecture of FM0/Manchester Encoding Technique for DSRC

Author1
Y. N. DURGA DEVI
Designation: Student
Mandava Institute of Engineering and Technology's
JAGGAYYAPET

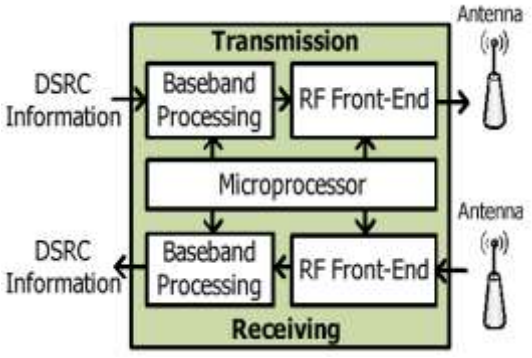
Author2
M. SUSEELA
Designation: guide
Mandava Institute of Engineering and Technology's
JAGGAYYAPET

Author3
B. SUBHAKARA RAO
Designation: HOD
Mandava Institute of Engineering and Technology's
JAGGAYYAPET

Abstract- The dedicated short-range communication (DSRC) is an evolving technology to push the smart transport system into our daily life. The DSRC technique usually adopt FM0 and Manchester encodings to reach dc-balance, boosting the signal quality. Nevertheless, the coding coordination between the FM0 and Manchester codes limits the potential to design a fully used VLSI architecture for Fm0 and Manchester. In this project, the (SOLS) technique is presented to overcome this drawback. The SOLS technique improves the hardware consumption rated from 57.14% to 100% for FM0 and Manchester codes. The projected method is capable of generating Manchester and Fm0 coding for DSRC applications.

INTRODUCTION

The dedicated short-range comm (DSRC) is a protocol for one or two way short range communication specially suited for intelligent transport systems. The DSRC system can be categorised into two types. Automobile to automobile communication and automobile to roadside communication. In automobile to automobile, the DSRC enable the data sending, data receiving and broadcasting among automotives for security issue and public information. The security issues consists of blind-spot, junction warning, distance from the other automotive, and collision-alarm. The automobile to roadside focuses on the smart transport service, such as ETC system (electronic toll collection). With ETC, the toll collection is achieved with the contact less IC card platform. Moreover, the ETC can be extended to the payment for parking-service, and gas refilling. Thus, the DSRC system plays an important role in modern automobile industry. Architecture of DSRC transceiver is elaborated in the following Figure. The top and lower parts are dedicated for broadcast and reception, respectively. This transceiver is again sub divided into three basic blocks. microprocessor, baseband processing, and RF front end. The microprocessor fetches the instructions to programme the tasks of base-band processing and RF-front end. The baseband processing carries out the modulation, error correcting, clock sync and encoding. The RF front end sends and accepts the wireless signal through the antenna.



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 front-end. 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. 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 codec in hardware realization.

B. Features of This Paper1

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.

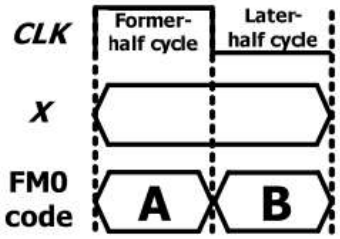


Fig. 2. Codeword structure of FM0.

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, areacompact 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 thereby the logic-1 is changed

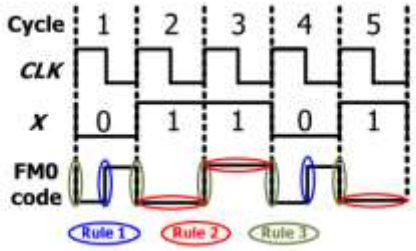


Fig. 3. Illustration of FM0 coding example.

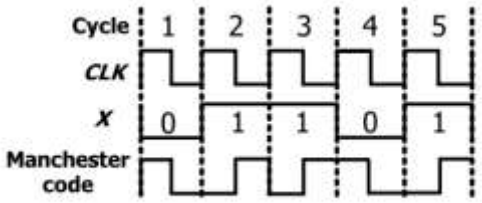


Fig. 4. Illustration of Manchester coding example.

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

$$X \oplus \text{CLK}. \qquad (1)$$

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.

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

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 03
March 2017

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. If the X is logic-1, the state transition must follow both rules 2 and 3. The only one next-state that can satisfy both rules for the X of logic-1 is S_4. Thus, the state-transition of each state can be completely constructed.

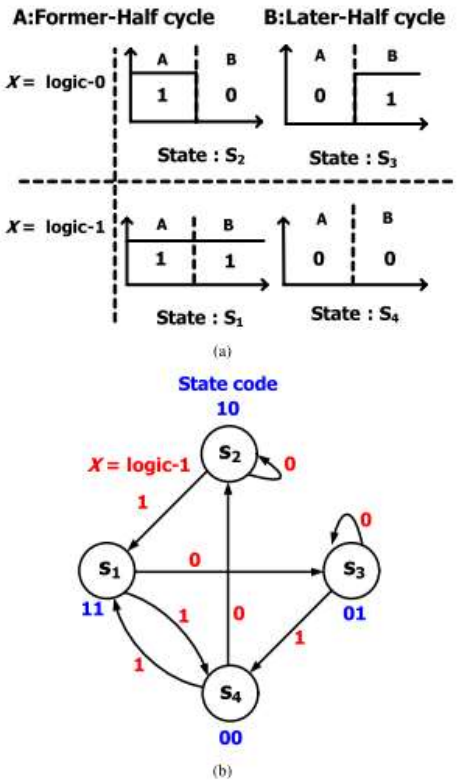


Fig. 5. Illustration of FSM for FM0. (a) States definition. (b) FSM of FM0.

TABLE II
TRANSITION TABLE OF FM0

Previous-state		Current-state			
		$A(t)$		$B(t)$	
$A(t-1)$	$B(t-1)$	$X=0$	$X=1$	$X=0$	$X=1$
1	1	0	0	1	0
1	0	1	1	0	1
0	1	0	0	1	0
0	0	1	1	0	1

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

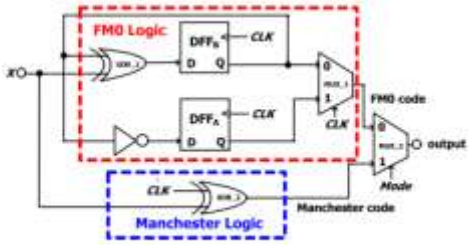


Fig. 6. Hardware architecture of FM0 and Manchester encodings.

TABLE III
HUR OF FM0 AND MANCHESTER ENCODINGS

Coding	Active components (transistor count) / Total components (transistor count)	HUR
FM0	6 (86) / 7 (98)	85.71%
Manchester	2 (26) / 7 (98)	28.57%
Average	4 (56) / 7 (98)	57.14%

of $A(t)$ and $B(t)$ are given as $B\,A((t\,t)) = = X\,B(\oplus\ t - B1\ (t)\ (\ -\ 1).\ (3\ 2))$ With both $A(t)$ and $B(t)$, the Boolean function of FM0 code is denoted as CLK $A(t)$ + CLK $B(t)$. (4) With (1) and (4), the hardware architectures of FM0 and Manchester encoders are shown in Fig. 6. The top part is the hardware architecture of FM0 encoder, and the bottom part is the hardware architecture of Manchester encoder. As listed in (1), the Manchester encoder is as simple as a XOR operation for X and CLK. Nevertheless, the FM0 encoding depends not only on the X but also on the previous-state of the FM0 code. The DFFA and DFFB store the state code of the FM0 code. The MUX -1 is to switch $A(t)$ and $B(t)$ through the selection of CLK signal. Both $A(t)$ and $B(t)$ are realized by (2) and (3), respectively. 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 = Active components / Total components × 100%. (5)

The component is defined as the hardware to perform a specific logic function, such as AND, OR, NOT, and flip flop. The active components mean the components that work for FM0 or Manchester encoding. The total components are the number of components in the entire hardware architecture no matter what encoding method is adopted. The HUR

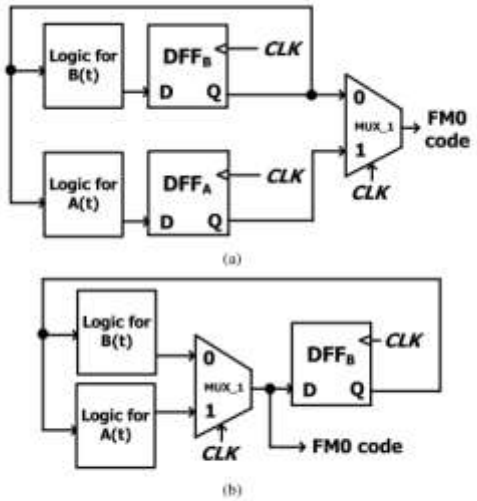


Fig. 7. Illustration of area-compact retiming on FM0 encoding architecture. (a) FM0 encoding without area-compact retiming. (b) FM0 encoding with area-compact retiming.

of FM0 and Manchester encodings is listed in Table III. For both encoding methods, the total components are 7, including MUX-2 to indicate which coding method is activated. For FM0 encoding, the active components are 6, and its HUR is 85.71%. For Manchester encoding, the active components are 2, comprising XOR-2 and MUX-2, and its HUR is as low as 28.57%. On average, this hardware architecture has a poor HUR of 57.14%, and almost half of total components are wasted. The transistor count of the hardware architecture without SOLS technique is 98, where 86 transistors are for FM0 encoding and 26 transistors are for Manchester coding. On average, only 56 transistors can be reused, and this is consistent with its HUR. 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 DFF$_A$ and DFF$_B$. According to (2) and (3), the transition of state code only depends on $B(t-1)$ instead of both $A(t-1)$ and $B(t-1)$. Thus, the FM0 encoding just requires a single 1-bit flip-flop to store the $B(t-1)$. If the DFF$_A$ is directly removed, a non-synchronization between $A(t)$ and $B(t)$ causes the logic fault of FM0 code. To avoid this logic fault, the DFF$_B$ is relocated right after the MUX-1, as shown in Fig. 7(b), where the DFF$_B$ 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.

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 03
March 2017

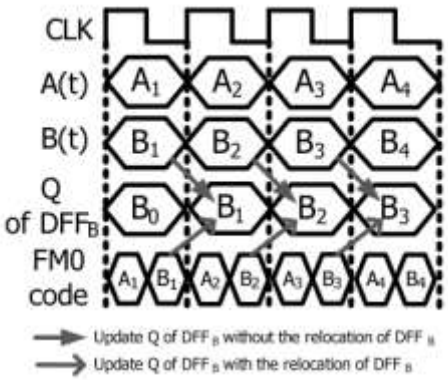


Fig. 8. Timing diagram of area-compact retiming for FM0 encoding.

TABLE IV
TRANSISTOR COUNT OF FM0 ENCODING ARCHITECTURE WITH AREA-COMPACT RETIMING

	Without area-compact retiming	With area-compact retiming
PMOS	36	25
NMOS	36	25
Total	72	50

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 DFF*B* 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 DFF*B*. Then, the upcoming positive-edge of CLK updates it to the *Q* of DFF*B*. As shown in Fig. 8, the timing diagram for the *Q* of DFF*B* is consistent whether the DFF*B* 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.

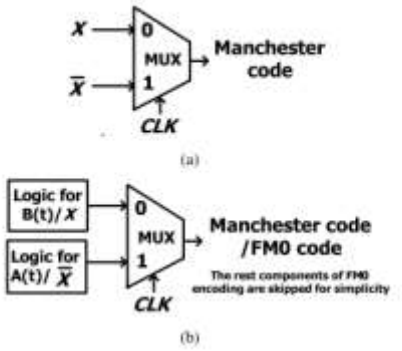


Fig. 9. Concept of balance logic-operation sharing for FM0 and Manchester encodings. (a) Manchester encoding in multiplexer. (b) Combines the logic-operations of Manchester and FM0 encodings.

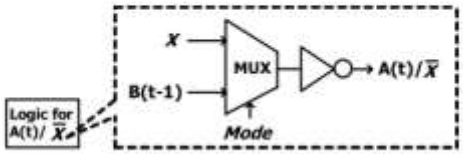


Fig. 10. Balance logic-operation sharing of $A(t)$ and $\overline{X}$.

B. Balance Logic-Operation Sharing

As mentioned previously, the Manchester encoding can be derived from $X \oplus$ CLK, and it is also equivalent to $X \oplus$ CLK = X CLK + X CLK. (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. The *A(t)* can be derived from an inverter of *B(t - 1)*, and *X* is obtained by an inverter of *X*. The logic for *A(t)/X* can share the same inverter, and then a multiplexer is placed before the inverter to switch the operands of *B(t - 1)* and *X*. The Mode indicates either FM0 or Manchester encoding is adopted. The similar concept can be also applied to the logic for *B(t)/ X*, as shown in Fig. 11(a). Nevertheless, this architecture exhibits a drawback that the XOR is only dedicated for FM0 encoding, and is not shared with Manchester encoding. Therefore, the HUR of this architecture is certainly limited. The *X* can be also interpreted as the $X \oplus 0$, and thereby the XOR 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

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 03
March 2017

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)

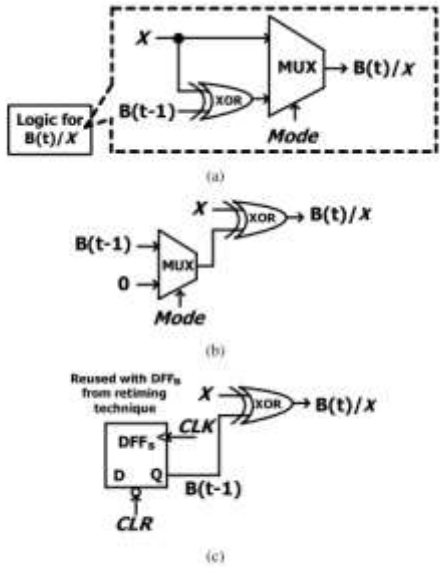


Fig. 11. Balance logic-operation sharing of $B(t)$ and X. (a) Without the XOR sharing. (b) With XOR sharing. (c) Sharing of the reused DFF$_B$ from area-compact retiming technique.

can be functionally integrated into the relocated DFF*B* from area-compact retiming technique, as shown in Fig. 11(c). The CLR is the clear signal to reset the content of DFF*B* to logic-0. The DFF*B* 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 DFF*B*. Hence, the multiplexer in Fig. 11(b) can be totally saved, and its function can be completely integrated into the relocated DFF*B*. The proposed VLSI architecture of FM0/Manchester encoding using SOLS technique is shown in Fig. 12(a). The logic for *A(t)/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)/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)/X* further incorporates an inverter in the series of MUX -2. This unbalance computation time between *A(t)/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)/X*. This shared inverter is relocated backward to the output of MUX-1. Thus, the logic computation time between *A(t)/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

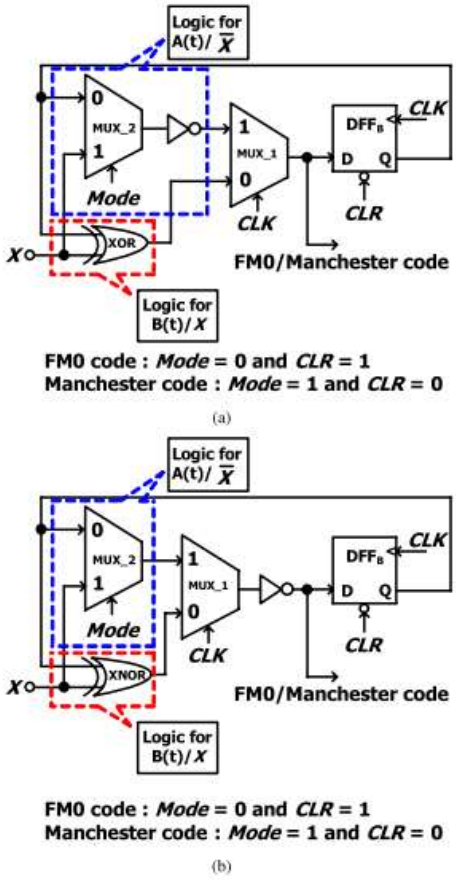


Fig. 12. VLSI architecture of FM0 and Manchester encodings using SOLS technique. (a) Unbalance computation time between $A(t)/\overline{X}$ and $B(t)/X$. (b) Balance computation time between $A(t)/\overline{X}$ and $B(t)/X$.

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 encodings. Therefore, the HUR of the proposed VLSI architecture is greatly improved.

International Journal of Research
Available at https://edupediapublications.org/journals
p-ISSN: 2348-6848
e-ISSN: 2348-795X
Volume 04 Issue 03
March 2017

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 Manchester and FM0 encoders, their critical paths are not identical. For Manchester encoding, the delay time is given as $T_{Man} = \max\{T_{MUX}, T_{XNOR}\} + T_{MUX} + T_{INV}$ (7) where T_{Man} denotes the delay time of Manchester encoding. The T_{MUX}, T_{XNOR}, and T_{INV} 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_{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 $T_{FM0} = T_{Man} + T_{DFF}$ (8) where the T_{FM0} is the delay time of FM0 encoding, and the T_{DFF} 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_{Man} not only dominates the timing of Manchester encoding, but also affects that of FM0 encoding. Hence, the logic components inside T_{Man} should be carefully considered in their implementation. If these logic components are totally designed with static CMOS, the T_{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 twoinput XNOR are shown in Fig. 13(a) and (b), respectively. The pull-down network of two-input multiplexer includes $M1$, $M2$, $M3$, and $M4$. The $M1$ and $M2$ are in parallel, and so are $M3$ and $M4$. 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 $T_{MUX\text{-}SC} = T_{INV} + C_A R + C_B 2R$ (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 capacitances of $M1$, $M2$, $M3$, and $M4$. The C_B gathers the junction capacitances of $M3$, $M4$, $M5$, and $M6$. Similarly, the propagation delay of static CMOS two-input XNOR is given as $T_{XNOR\text{-}SC} = T_{INV} + C_C R + C_D 2R$. (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 $T_{MUX\text{-}SC}$ is greater than $T_{XNOR\text{-}SC}$.

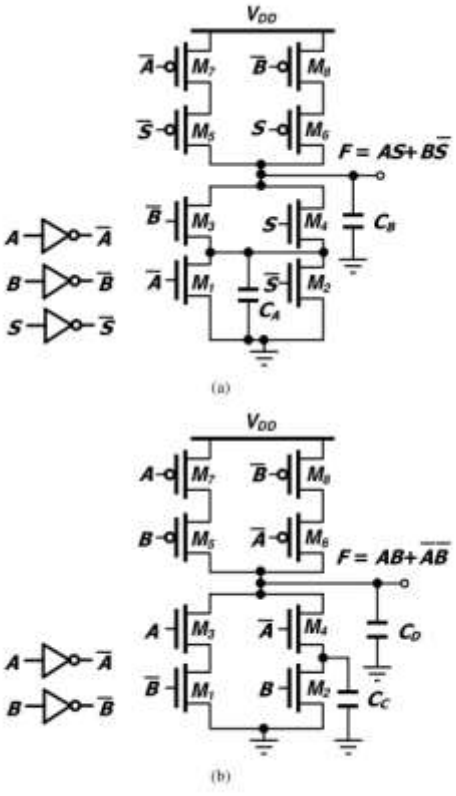


Fig. 13. Static CMOS topologies of multiplexer and XNOR. (a) Two-input multiplexer. (b) Two-input XNOR.

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 $2T_{MUX\text{-}SC} = 2(T_{INV} + C_A R + C_{B2} R)$. (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 $T_{MUX\text{-}TG} = RC_E$ (12) where the C_E aggregates the junction capacitances of $M1$, $M2$, $M3$, and $M4$. The R stands for the equivalent resistances of $M1$ and $M2$ in parallel, or $M3$ and $M4$ in parallel. Similarly, the propagation delay of transmission-gate two-input XNOR is given as $T_{XNOR\text{-}TG} = T_{INV} + RC_F$ (13) where the C_F gathers the junction capacitances of $M1$, $M2$, $M3$, and $M4$. The $T_{XNOR\text{-}TG}$ causes slightly longer propagation delay than $T_{MUX\text{-}TG}$. 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 $T_{XNOR\text{-}TG} + T_{MUX\text{-}TG} = T_{INV} + RC_E + RC_F$. (14)

CONCLUSION

This project presents a fully re used VLSI architecture of Fm0 and Manchester encodings and Miller encodings. The proposed encoding techniques are used for several DSRC (Dedicated Short Range Communication). Fully re used VLSI hardware means that every elements of the circuit architecture is in active state while generating different encodings and results in high efficiency design with less power consumption as every element in the architecture is used everytime when the circuit is in opertating mode. The simulations are done in Modelsim 6.5b tool and simulation of output waveforms are verified for encoding techniques of Manchester and FM0.

REFERENCES

[1] F. Ahmed-Zaid, F. Bai, S. Bai, C. Basnayake, B. Bellur, S. Brovold, *et al.*, "Vehicle safety communications—Applications (VSC-A) final report," U.S. Dept. Trans., Nat. Highway Traffic Safety Admin., Washington, DC, USA, Rep. DOT HS 810 591, Sep. 2011.

[2] J. B. Kenney, "Dedicated short-range communications (DSRC) standards in the United States," *Proc. IEEE*, vol. 99, no. 7, pp. 1162–1182, Jul. 2011.

[3] J. Daniel, V. Taliwal, A. Meier, W. Holfelder, and R. Herrtwich, "Design of 5.9 GHz DSRC-based vehicular safety communication *IEEE Wireless Commun. Mag.*, vol. 13, no. 5, pp. 36–43, Oct. 2006.

[4] 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.

[5] A. Karagounis, A. Polyzos, B. Kotsos, and N. Assimakis, "A 90nm 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.

[6] 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. Intell. Inf. Hiding Multimedia Signal Process.*, Sep. 2009, pp. 538–541.

[7] 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.

[8] M. A. Khan, M. Sharma, and P. R. Brahmanandha, "FSM based FM0 and Miller encoder for UHF RFID tag emulator," in *Proc. IEEE Adv. Comput. Conf.*, Mar. 2009, pp. 1317–1322.

[9] 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. Adv. Commun. Technol.* Jan. 2013, pp. 98–103.

[10] I.-M. Liu, T.-H. Liu, H. Zhou, and A. Aziz, "Simultaneous PTL buffer insertion and sizing for minimizing Elmore delay," in *Proc. Int. Workshop Logic Synth.*, May 1998, pp. 162–168.

[11] H. Zhou and A. Aziz, "Buffer minimization in pass transistor logic," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.*, vol. 20, no. 5, pp. 693–697, May 2001.

[12] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A Systems Perspective*, 2nd ed., Upper Saddle River, NJ, USA: Pearson Educ. Ltd., 1993, pp. 98–103.