



DESIGN OF AREA EFFICIENT BINARY TO GRAY CODE CONVERTER USING MENTOR GRAPHICS

Sowmya Bhat¹, Avinash N.J², Kusuma Prabhu³, Rajashree Nambiar⁴

Assistant Professor, Department of Electronics & Communication Engineering,

Shri Madhwa Vadiraja Institute of Technology & Management, Bantakal

Email:sowmyabhat10@gmail.com¹,

avinash.yuvaraj@gmail.com²,kusum.prabhu@gmail.com³,raji24oct@gmail.com⁴

Abstract

Moore's law explains the requirement of the transistors for VLSI design. It gives the empirical observation that transistor density and performance of integrated circuits, doubles every year, which was then revised to doubling every two years. Unfortunately, such performance improvements have been accompanied by an increase in power and energy dissipation of the systems. Higher power and energy dissipation in high performance systems require more expensive packaging and cooling technologies, increase cost, and decrease system reliability. Nonetheless, the level of on-chip integration and clock frequency will continue to grow with increasing performance demands, and the power and energy dissipation of high-performance systems will be a critical design constraint. The design of code converters which forms the basic building blocks of all digital VLSI circuits has been undergoing a considerable improvement, being motivated by basic design goals, viz. minimizing the transistor count, minimizing the power consumption. The XOR gates form the fundamental building block of code converters. Enhancing the performance of the XOR gates can significantly improve the performance of the code converters.

Index Terms: About four key words or phrases in alphabetical order, separated by commas.

I. INTRODUCTION

Different types of XOR gates that have been realized over the years. The code converters are more complex and power consuming circuits in digital design. To reduce the power dissipation several code converters are designed but they are not suitable for operation in the sub threshold region. These designs require more transistors and lead to increase in area, so these are not suitable for small and low price systems. To reduce the power and area requirements of the computational complexities, the size of transistors are shrunk into the deep sub-micron region and predominantly handled by process engineering. Many design architecture and techniques have been developed to reduce power dissipation. The power consumption techniques are CMOS complementary logic, Pseudo NMOS, Dynamic CMOS, Clocked CMOS logic (C²MOS), CMOS Domino logic, Cascade voltage switch logic (CVSL), Modified Domino logic, Pass Transistor Logic (PTL). These designs require a large number of transistors, resulting in a large area, not suitable for small, low-priced systems.

The most useful low power consumption technique is PTL. The PTL advantages are:

- High speed, due to small node capacitances.
- Low power dissipation, as a result of reduced number of transistors.
- Lower interconnection effect, due to smaller area.

There are two main drawbacks in PTL:

- The threshold voltage across the single channel pass transistors results in reduced drive and hence slower operation at reduced voltages.
- The high input voltage level is not VDD the PMOS device in inverter is not fully turned off.

In order to overcome these drawbacks we use Transmission Gate (TG) logic.

The main advantage of the TG logic is:

- The complex logic functions are implemented by using small number of transistors.
- The logic level swing can be reduced by using PTL.

The combination of NMOS PT with CMOS output inverters is called Complementary pass transistor logic (CPL). It suffers from the static power and low swing at gates of the output inverters. To reduce the static power dissipation and full swing operation we use the Double pass transistor logic (DPL). Double pass transistor logic (DPL) has more area due to presence of PMOS transistor.

II. BINARY AND GRAY CODE

A binary code is a way of representing text or computer processor instructions by the use of the binary number system's two-binary digits 0 and 1. This is accomplished by assigning a bit string to each particular symbol or instruction. For example, a binary string of eight binary digits (bits) can represent any of 256 possible values and can therefore correspond to a variety of different symbols, letters or instructions. In computing and telecommunication, binary codes are used for any of a variety of methods of encoding data, such as character strings, into bit strings. Those methods may be fixed-width or variable-width. In a fixed-width binary code, each letter, digit, or other character, is represented by a bit string of the same length; that bit string, interpreted as a binary number, is usually displayed in code tables in octal, decimal or hexadecimal notation.

Another code worth mentioning at this point is the *Gray code* (or *reflective-binary*) which was invented by Elisha Gray in 1878 and later re-invented by Frank Gray in 1949.

Although it is rarely used in computer arithmetic, it has some useful properties which

make it attractive to A/D conversion. Notice that in Gray code, as the number value changes, the transitions from one code to the next involve only one bit at a time. Contrast this to the binary code where all the bits change when making the transition between 0111 and 1000. Some ADCs make use of it internally and then convert the Gray code to a binary code for external use. The reflected binary code was originally designed to prevent spurious output from electromechanical switches. Today, Gray codes are widely used to facilitate error correction in digital communications such as digital terrestrial television and some cable TV systems. Patent applications give "Gray code" as an alternative name for the "reflected binary code". One of those also lists "minimum error code" and "cyclic permutation code" among the names. The problem with natural binary codes is that, with real (mechanical) switches, it is very unlikely that switches will change states exactly in synchrony. In the transition between the two states shown above, all three switches change state. In the brief period while all are changing, the switches will read some spurious position. Even without key bounce, the transition might look like 011 — 001 — 101 — 100. When the switches appear to be in position 001, the observer cannot tell if that is the "real" position 001, or a transitional state between two other positions. If the output feeds into a sequential system (possibly via combinational logic) then the sequential system may store a false value. The reflected binary code solves this problem by changing only one switch at a time, so there is never any ambiguity of position. One of the earliest practical ADCs to use the Gray code was electron beam encoder developed by Bell Labs. The basic electron beam coder concepts are shown in Figure 3.1 for a 4-bit device. The early tubes operated in the serial mode as shown in Figure 1 (a). The analog signal is first passed through a sample-and-hold, and during the "hold" interval, the beam is swept horizontally across the tube. The Y-deflection for a single sweep therefore corresponds to the value of the analog signal from the sample-and-hold. The shadow mask is coded to produce the proper binary code, depending on the vertical deflection. The code is registered by the collector, and the bits are generated in serial format. Later tubes used a fan-shaped beam as

shown in Figure 1 creating a "flash" converter delivering a parallel output word.

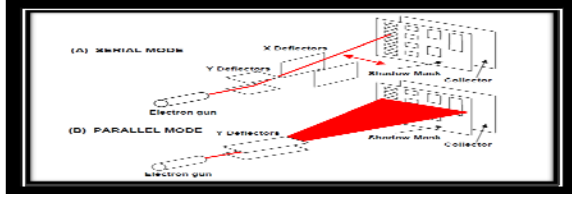


Figure 1: The Electron Beam Coder: (a) Serial Mode and (b) Parallel or "Flash" Mode

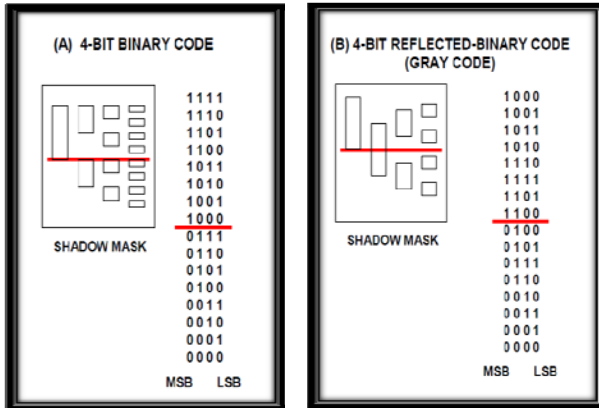


Figure 2: Electron Beam Coder Shadow Masks for Binary Code (A) and Gray Code (B)

The errors associated with binary shadow masks and how they were eliminated by using a Gray code shadow mask as shown in Figure 3 (a and b). As mentioned above, the Gray code has the property that adjacent levels differ by only one digit in the corresponding Gray-coded word. Therefore, if there is an error in a bit decision for a particular level, the corresponding error after conversion to binary code is only one least significant bit (LSB). In the case of mid-scale, note that only the MSB changes. It is interesting to note that this same phenomenon can occur in modern comparator-based flash converters due to comparator metastability. With small overdrive, there is a finite probability that the output of a comparator will generate the wrong decision in its latched output, producing the same effect if straight binary decoding techniques are used. In many cases, Gray code, or "pseudo-Gray" codes are used to decode the comparator bank. The Gray code output is then latched, converted to binary, and latched again at the final output.

Other examples where Gray code is often used in the conversion process to minimize errors are shaft encoders (angle-to-digital) and optical encoders. ADCs which use the Gray code internally almost always convert the Gray code output to binary for external use. The conversion from Gray-to-binary and binary-to-Gray is easily accomplished with the exclusive-or logic function as shown in Figure 3.

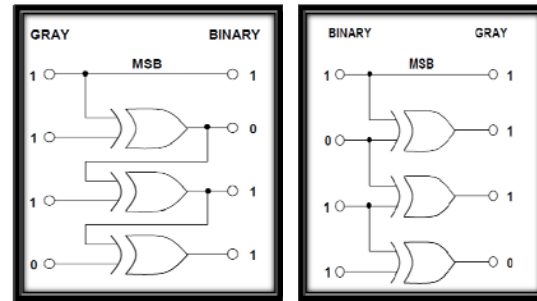


Figure 3: Binary-to-Gray and Gray-to-Binary Conversion Using the XOR

More formally, a Gray code is a code assigning to each of a contiguous set of integers, or to each member of a circular list, a word of symbols such that each two adjacent code words differ by one symbol. These codes are also known as single-distance codes, reflecting the Hamming distance of 1 between adjacent codes. There can be more than one Gray code for a given word length, but the term was first applied to a particular binary code for the non-negative integers, the binary-reflected Gray code, or BRGC, the four-bit version of which is shown above. Design of Gray to Binary Code using PMOS transistors is as shown in the Figure 4.

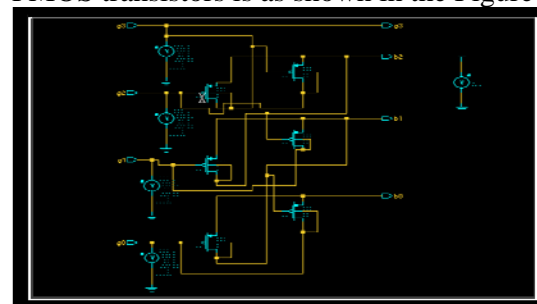


Figure 4: Design of Gray to Binary Code two PMOS transistors

III. DESIGN IMPLEMENTATION USING MENTOR GRAPHICS

The basic building gate of Gray to Binary code converter is EX-OR. The early designs of XOR

gates were based on either four transistors or three transistors that are conventionally used in most designs over the last decade. The previous designs of the four transistors and three transistors are shown in figure 5_10. The proposed two transistor XOR gates can be designed using general logic implementation.

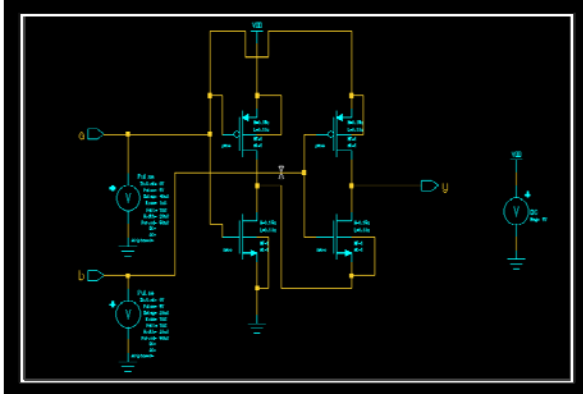


Fig.5: Design of 4-T XOR (Type 1)

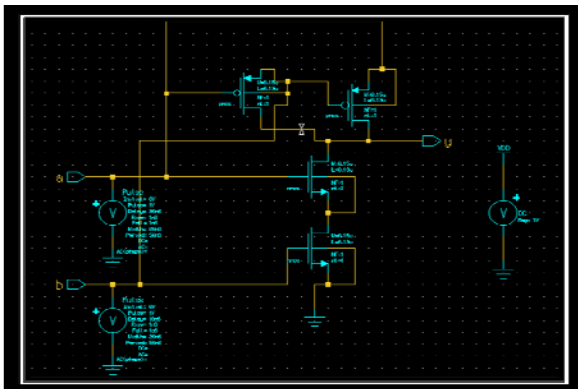


Fig. 6: Design of 4-T XOR (Type 2)

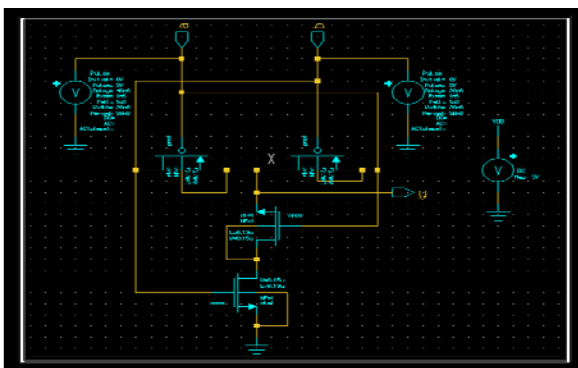


Fig. 7: Design of 4-T XOR (Type 3)

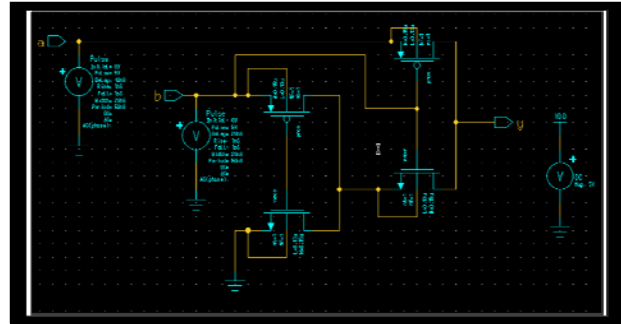


Fig. 8: Design of 4-T XOR (Type 4)

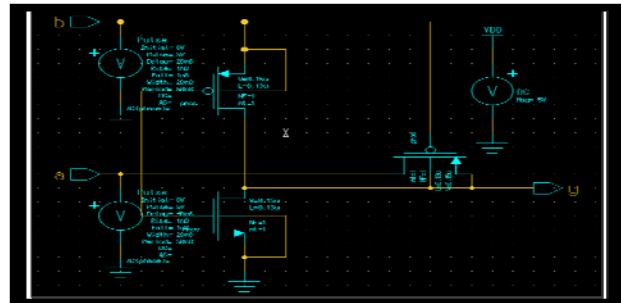


Fig. 9 Design of 3-T XOR

XOR gates designed as shown in the Fig 3.1 to 3. 5 are not suitable for operation in the sub threshold region. These designs require more transistors and lead to increase in area, so these are not suitable for small and low price systems. To overcome this disadvantage a two transistor XOR gate is proposed in this paper. The circuit operation of proposed design of 2T -XOR is as follows, when $A=0$ and $B=0$ both the PMOS transistors are ON and it will produce the output as low. When either one of the transistor is ON, it produces output as high, when $A=1$ and $B=1$ both the PMOS transistors are OFF and it will produce the output as low. The design and implementation of EX-OR gate using two PMOS transistors is dissipates less power and it requires less area.

The design of a two transistor XOR gate is shown in figure 10.

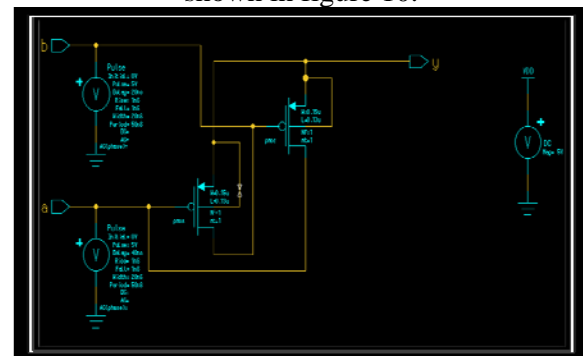


Fig.10: Proposed Design of 2T -XOR

The digital VLSI circuits has undergone a considerable improvement, being motivated by basic design goals, viz. minimizing the transistor count, minimizing the power consumption as shown in the proposed design of two transistor XOR gate .As the XOR gates form the fundamental building block of code converters, enhancing the performance of the XOR gates can significantly improve the performance of the code converters.

IV. RESULT AND SIMULATION

In Mentor Graphics Tool both EX-OR and Gray to Binary code converter are designed. After that these designs were simulated. By using simulation results the input and output simulated waveforms are obtained as shown in Figure 11-15. for the previous designs of the four transistors and three transistors and as shown in Figure 5.6 for a two transistor XOR gate.

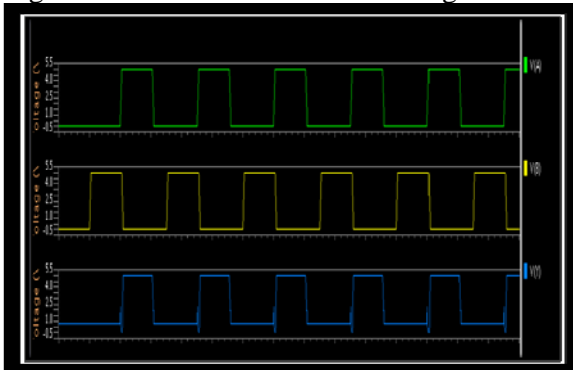


Fig. 11 Simulated waveforms of 4-T XOR (Type 1)

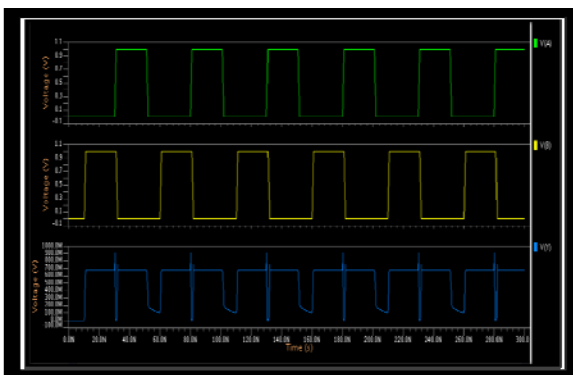


Fig. 12 Simulated waveforms of 4-T XOR (Type 2)

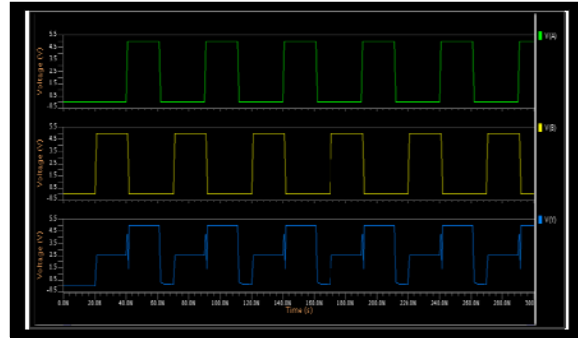


Fig.13 Simulated waveforms of 4-T XOR (Type 3)

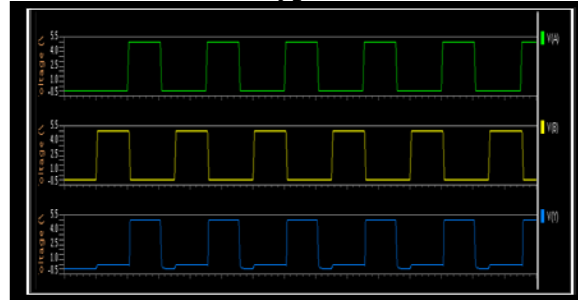


Fig.14 Simulated waveforms of 4-T XOR (Type 4)

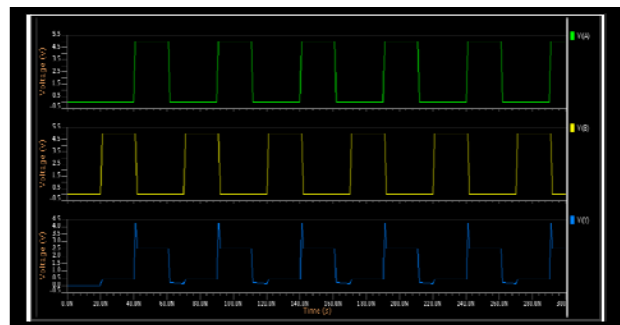


Fig. 15 Simulated waveforms of 3-T XOR

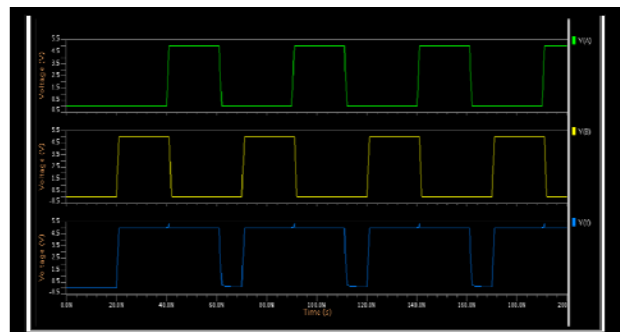


Fig. 16: Simulated waveforms for a two transistor XOR gate

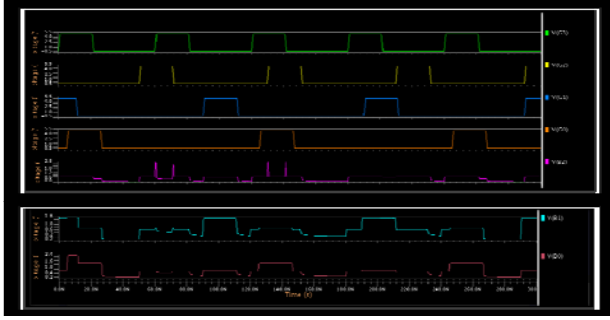


Fig. 17: Simulated waveforms for Gray to Binary Code two PMOS transistors

The transistor count for the XOR gate and the Binary to Gray code converter, obtained from the schematic representation is as shown in Table 1.

Circuit Transistors	No. of
Binary to Gray	6
EX-OR	2

Table. 1 Comparison of Code Converters (Number of Transistors)

From the simulated waveform it is clear that the proposed 2T XOR gate gives a better performance when compared to the previous designs using 4T and 3T.

Figure 17 represent the simulated waveforms for Gray to Binary Code using a novel two PMOS transistors.

V. CONCLUSION

A novel approach, using two PMOS transistors technique has been adopted for implementation of area efficient circuit. The implementation of Gray to Binary Code Converter has been presented in two PMOS transistors are four and can be extended to other codes.

The future scope may include integration of the proposed two PMOS transistors technique design of Gray to Binary Code in complex digital systems, digital communication systems and telecommunications. The implementation of Gray to Binary Code Converter has been presented in two PMOS transistors are four and can be extended to other codes.

REFERENCES

- [1] Pakkiraiah Chakali, Design of High Speed Six Transistor Full Adder using a Novel Two Transistor XOR Gates, IJARCSEE ISSN: 2277 – 9043.
- [2] Pakkiraiah Chakali, A Novel Low power and Area efficient Carry Look Ahead Adder Using GDI Technique, IJRCET
- [3] 9T Full Adder Design in Subthreshold Region, Shiwani Singh, Tripti Sharma, K. G. Sharma, and B. P. Singh
- [4] N. Weste and K. Eshraghian Principles of CMOS digital design.
- [5] Fundamentals of Sampled Data Systems, Chapter2.