Journal of Equity in Science and Sustainable Development

Vol. 1(1), pp. 49-55, June 2017

http://www.mwu.edu.et
Email: jessd@mwu.edu.et

Full Length Research Paper

Performance Analysis of Error Detection and Correction Techniques for Communication Modules

Jennifer Peter[1], S.Savarirani[2] and S.Ramasamy[3]

[1]Engineer-STA, Mbit wireless, email:jenicool.tech@gmail.com
[2]Assistant Professor, RMD Engineering College, email: savarirani@gmail.com
[3]Professor Electrical, College of Electrical and Mechanical Engineering, Addis Ababa Science and Technology University, Kilinto, Addis Ababa, email: rams@aastu.edu.et

Received: 16 March, 2017; Accepted: 13 June, 2017; Published: 26 June, 2017

ABSTRACT

The error detection and error correction techniques play a major role in the digital data transformation over reliable communications. They are suitable techinques to implement them in deferrent signal conditions in the digital communication. In this work, the various error detection and error correction methods were analyzed based on power and area. The design was simulated and synthesized using synopsys verilog compiler simulator (VCS) and cadence register transfer language (RTL) compiler and cadence encounter too which were used for application specific integrated circuit (ASIC) implementation.

Keywords: *Cyclic Redundancy Code, Error detection, Hamming Code, Parity Checker*

Corresponding author *address: email:* savarirani@gmail.com

Introduction

Environmental interference and physical defects can cause random bit errors during data transmission in digital communication systems. Digital data is transmitted over a channel and the channel consists of noise. This noise may distort the messages to be sent and the receiver receives may not be the same as what the sender sends. Error coding is a method of detecting and correcting these errors in a wide range of communication systems especially in computer memory, optical data storage media, satellite, network communications, cellular telephone networks, deep space communications and almost all digital data communication. Error like SEU (single event upset) and stuck at fault occurs due to the radiation into the environment, this cause message bit to flip i.e. from 1 to 0 or 0 to 1 in the memory. This error is also called as temporary error or soft error. This study was conducted to analyze various error detection and correction methods based on power and area.

Error Detection And Correction Techniques/ Methodology

Data transmitted over a communication channel is not completely an error free. The changes to the data are caused due to external interference, radiation, signal distortion, attenuation or from noise (Vishal and Amit, 2014). There are two types of errors (Martin and Steve, 2011). Primarily single error in which only one bit is changed and secondly the burst error in which more than one bit is changed. There are various error detection and correction techniques such as cyclic redundancy checks (CRC), parity check, and hamming code.

Parity Checker

In communications, parity checking refers to the use of parity bits to check that data has been transmitted accurately. The parity bit is added to every data unit that is transmitted. The parity bit for each unit is set so that all bytes have either an odd number or an even number of set bits (Thornton, 19997).

Table 1: Parity Checker

7 data bits	Count of bit 1	8 bit including parity bit	
		Even	odd
0000000	0	0000000**0**	0000000**1**
1010001	3	1010001**1**	1010001**0**
1100011	4	1100011**0**	1100011**1**
1111111	5	1111111**1**	1111111**0**

Table 1 shows the parity checker, it can be calculated via an XOR sum of the bits, yielding 0 for even parity and 1 for odd parity.

Cyclic Redundancy Check (CRC)

Cyclic redundancy check (CRC) is an error detecting code, which is used to identify corruption in the block of transmitted data or stored data. The linear feedback shifted register (LFSR) is the generic inexpensive hardware used for the CRC, which assumes serial data input. The used polynomial determines the capability of the error detection. The performance of the polynomial is affected by the data, its length as well as the anticipated error patterns (Martin and Steve, 2011). Different applications might favor different polynomials.

The register needs to be cleared initially in order to obtain the CRC. After the injection of the message and additional zeros, the specific CRC will be hold by the register. The same procedure can be applied to the receiver end to verify the received message with its appended CRC. The only difference is that the CRC will be shifted into the circuit instead of the zeros. The register finally becomes zero, if no error has been detected.

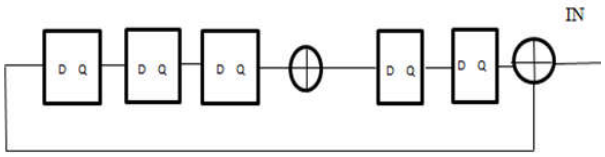


Figure 1: CRC using internal LFSR

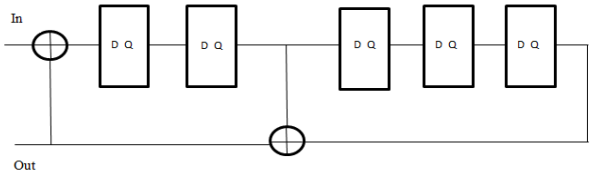


Figure 2: CRC using external LFSR

There are two types of LFSR like internal LFSR and external LFSR. Thus CRC can be implemented in both models as shown in Fig 1 and Fig 2. Implementing internal LFSR in CRC will be a GALOIS MODEL and external LFSR implementation will be on FIBONACCI model.

Hamming Code

Hamming code is an error correction code that can be used to detect single and double-bit errors and correct single-bit errors that can occur when binary data is transmitted from one device into another (Vishal and Amit, 2014).

Hamming Encoder

To calculate the numbers of redundant bits (r) required to detect the error, we have (d+r) as the total number of bits, which are to be transmitted; then r must be able to indicate at least d+r+1 different value (Vishal and Amit, 2014). Of these, one value means no error, and remaining d+r values indicate error location of error in each of d+r locations. So, d+r+1 state must be distinguishable by r bits, and r bits can indicate 2r states. Hence, 2r must be greater than d+r+1. The value of r must be determined by putting in the value of d in the relation. For example, if d is 7, then the smallest value of r that satisfies the above relation is 4. So the total bits, which are to be transmitted is 11 bits (d + r = 7 + 4 =11). (Fig 3) These redundancy bits are placed in positions 1, 2, 4 and 8 (the positions in an 11-bit sequence that are powers of 2). For clarity in the examples below, these bits are referred to as r1, r2, r3 and r4. In the Hamming code, each r bit is the parity bit for one combination of data bits as shown below:

r1: 1, 3, 5, 7, 9, 11
r2: 2, 3, 6, 7, 10, 11
r4: 4, 5, 6, 7
r8: 8, 9, 10, 11

11	10	9	8	7	6	5	4	3	2	1
d_6	d_5	d_4	r_8	d_3	d_2	d_1	r_4	d_0	r_2	r_1

Figure 3: Position of redundancy bits in hamming code

Hamming Decoder

The parity bits are recalculated at the receiver end shown in figure 4. The decimal value of the k parity bits provides the bit-position in error, if any (Ravi and Ashwini, 2014). The Hamming code is used for correction for 7-bit numbers (d7 d6 d5 d4 d3 d2 d1) with the help of four redundant bits (r4 r3 r2 r1) with error positions (C3 C2 C1 C0). For example, data 11111111000, first r1 is calculated considering the parity of the bit positions, 1, 3, 5, and 7,9,11. Secondly, the parity bits r2 is calculated considering bit positions 2, 3, 6, 7,10,11, then the parity bits r4 is calculated considering bit positions 4, 5, 6 and 7, finally the parity bits r8 is calculated considering bit positions 8,9,10,11 as shown. If any corruption occurs in any of the transmitted code 11111111000, the bit position in error can be found out by calculating r4 r3 r2 r1 at the receiving end. Fig 4 shows the Hamming decoder for finding error position. For example, if the received code word is

10111111000, the recalculated value of r4 r3 r2 r1 is 1010, which indicates that bit position in error is 10, the decimal value of 1010.

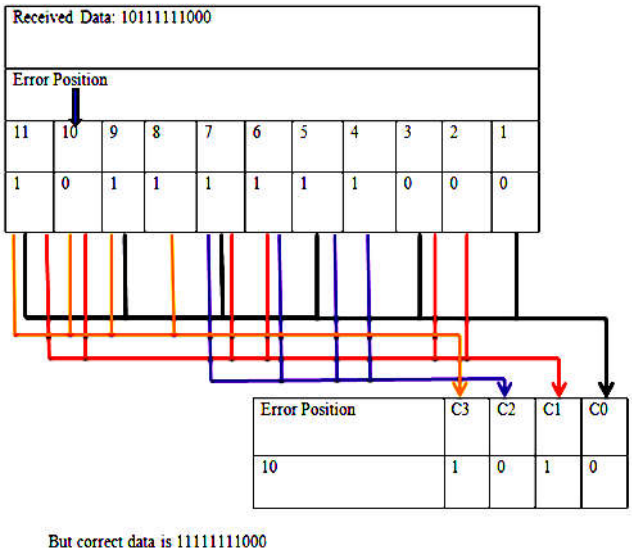


Figure 4: Hamming decoder for finding error position

Simulation Results And Analysis

To analyze the different error detection RTL code is verified and synthesized using Synopsys VCS and Cadence RTL compiler targeted to UMC90nm CMOS technology. Fig.5. shows the simulation result for CRC model.

After the simulation, various error detection techniques have been synthesized using Cadence RTL compiler. The designs are synthesized for constant timing slack and the optimized area and power results are obtained.

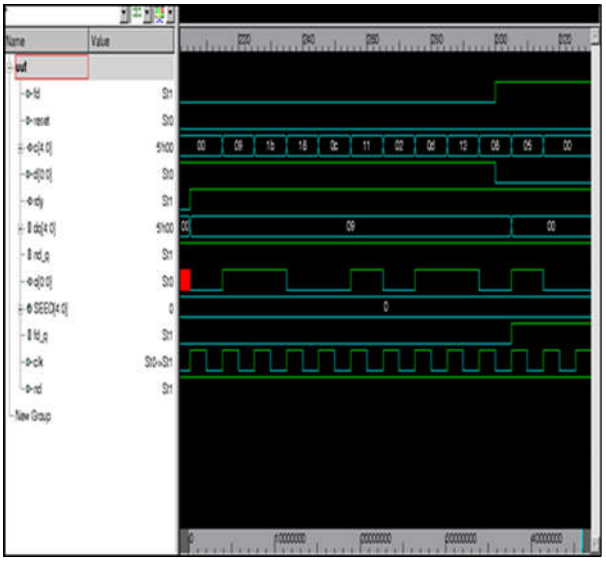


Figure 5: Waveform for CRC model

```
Global mapping status
=====================
                                  Group
                                 Tot Wrst
                          Total  Weighted
Operation                  Area   Slacks
-------------------------------------------------------------
 global_map                341       0

Global incremental target info
==============================
Cost Group 'clk' target slack:   156 ps
Target path end-point (Pin: lfsr_q_reg[2]/SI (SDFFSHQX1/SI))


Global incremental optimization status
======================================
                                  Group
                                 Tot Wrst
                          Total  Weighted
Operation                  Area   Slacks
-------------------------------------------------------------
 global_incr               337       0
```

Figure 6: Area report for CRC

```
                 Leakage    Dynamic     Total
Instance Cells Power(nW) Power(nW) Power(nW)
-------------------------------------------------
crc         49  1627.513 59955.729 61583.242
```

Figure 7: Power report for CRC

Fig 6 and 7 shows area and power report for CRC model. The timing slack of about 7224ps is maintained.

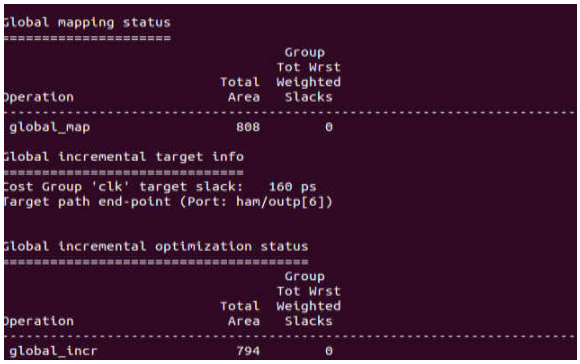


Figure 8. Area report for hamming code

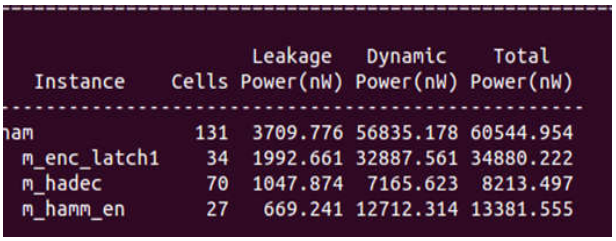
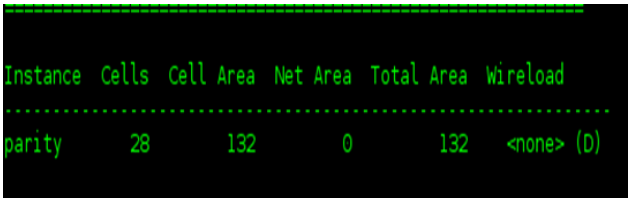


Figure 9. Power report for hamming code

For hamming encoder and decoder the timing slack is maintained 5719ps. Fig 8 and 9 shows area and power report for hamming model.



Figugre 10. Area report for parity checker

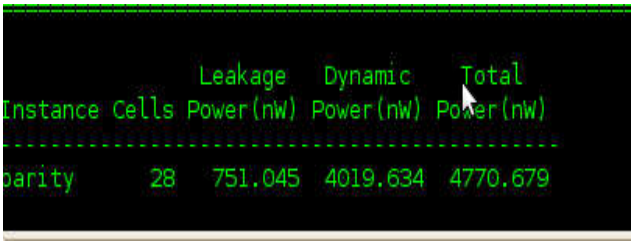


Figure 11. Power report for parity checker

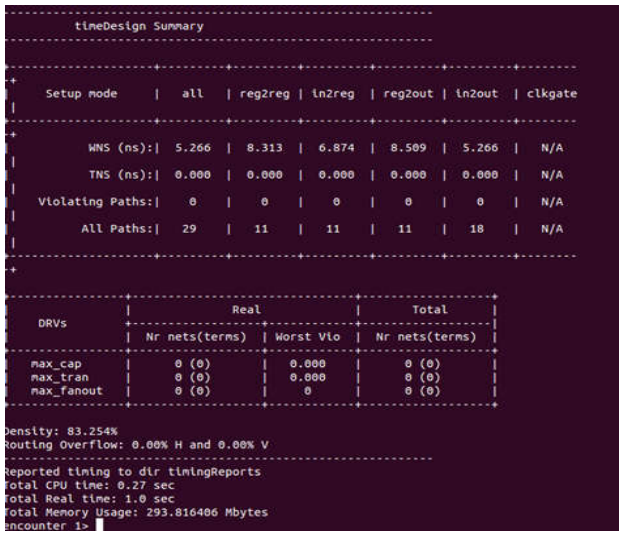
240ps is maintained for Parity checker. Fig 10 and 11 shows area and power report for parity checker. Table 2 represents the area and power report for various error detection and correction methods.

After the successful synthesis, the physical design using the cadence encounter has been done. This design involves the floorplaning, routing and generating a GDS II file. Generated net-list from the compiler is imported into cadence encounter. After loading corresponding LEF files and technology libraries, an automated floor plan is done with the suitable ratios.

Table 2. Area and power report

Error detection and correction techniques	Area (mm^2)	Leakage Power (nW)	Dynamic power (nW)
PARITY CHECKER	132	751.04	4019.64
CRC	337	1627.51	59955.73
HAMMING CODE	794	3709.78	56835.18

The core die is surrounded by power rings (VDD and VSS) after the floor planning. Furthermore, the horizontal and vertical power stripes across the dies are assigned. Now, the design macros are placed across the die so that optimum design can be achieved.



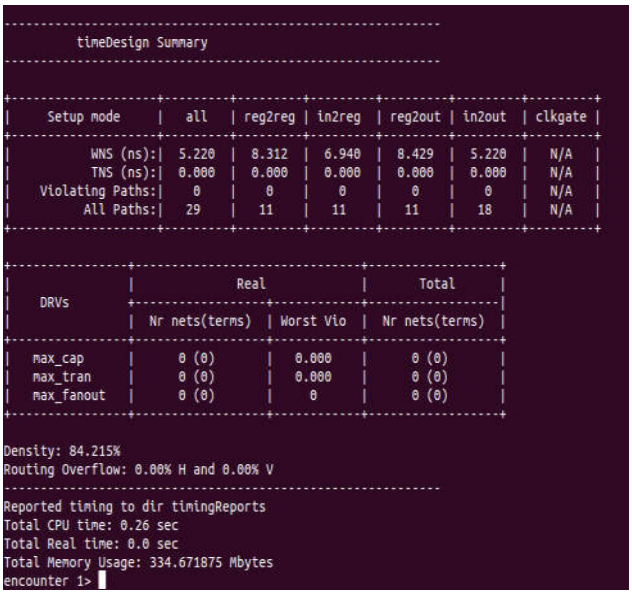
```
                    timeDesign Summary
--------------------------------------------------------

+--------------------+---------+---------+---------+---------+---------+--------+
|     Setup mode     |   all   | reg2reg | in2reg  | reg2out | in2out  | clkgate |
+--------------------+---------+---------+---------+---------+---------+--------+
|           WNS (ns):|  5.266  |  8.313  |  6.874  |  8.509  |  5.266  |   N/A  |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |  0.000  |  0.000  |   N/A  |
|    Violating Paths:|    0    |    0    |    0    |    0    |    0    |   N/A  |
|          All Paths:|   29    |   11    |   11    |   11    |   18    |   N/A  |
+--------------------+---------+---------+---------+---------+---------+--------+

+----------------+-------------------------------+------------------+
|                |              Real             |       Total      |
|    DRVs        +---------------+---------------+------------------|
|                | Nr nets(terms)| Worst Vio     | Nr nets(terms)   |
+----------------+---------------+---------------+------------------+
|   max_cap      |     0 (0)     |   0.000       |      0 (0)       |
|   max_tran     |     0 (0)     |   0.000       |      0 (0)       |
|   max_fanout   |     0 (0)     |     0         |      0 (0)       |
+----------------+---------------+---------------+------------------+

Density: 83.254%
Routing Overflow: 0.00% H and 0.00% V
------------------------------------------------------------
Reported timing to dir timingReports
Total CPU time: 0.27 sec
Total Real time: 1.0 sec
Total Memory Usage: 293.816406 Mbytes
encounter 1>
```

Figure 12. Pre-CTS timing report



```
                    timeDesign Summary
--------------------------------------------------------

+--------------------+---------+---------+---------+---------+---------+---------+
|     Setup mode     |   all   | reg2reg | in2reg  | reg2out | in2out  | clkgate |
+--------------------+---------+---------+---------+---------+---------+---------+
|           WNS (ns):|  5.220  |  8.312  |  6.940  |  8.429  |  5.220  |   N/A   |
|           TNS (ns):|  0.000  |  0.000  |  0.000  |  0.000  |  0.000  |   N/A   |
|    Violating Paths:|    0    |    0    |    0    |    0    |    0    |   N/A   |
|          All Paths:|   29    |   11    |   11    |   11    |   18    |   N/A   |
+--------------------+---------+---------+---------+---------+---------+---------+

+----------------+-------------------------------+------------------+
|                |              Real             |       Total      |
|    DRVs        +---------------+---------------+------------------|
|                | Nr nets(terms)| Worst Vio     | Nr nets(terms)   |
+----------------+---------------+---------------+------------------+
|   max_cap      |     0 (0)     |   0.000       |      0 (0)       |
|   max_tran     |     0 (0)     |   0.000       |      0 (0)       |
|   max_fanout   |     0 (0)     |     0         |      0 (0)       |
+----------------+---------------+---------------+------------------+

Density: 84.215%
Routing Overflow: 0.00% H and 0.00% V
------------------------------------------------------------
Reported timing to dir timingReports
Total CPU time: 0.26 sec
Total Real time: 0.0 sec
Total Memory Usage: 334.671875 Mbytes
encounter 1>
```

Figure13. Post-CTS timing report

Further clock tree synthesis (CTS) was done according to Guirong *et al. (*2009). The goal of clock tree synthesis (CTS) is to minimize skew and insertion delay. Pre-CTS and Post-CTS for both setup hold mode were carried out. In addition, optimization was carried out in case of negative slack. Fig.12 and 13 show the timing reports for Pre-CTS and Post-CTS in setup mode. Once the clock tree synthesis is done then the die is routed in optimum fashion. In Cadence Encounter, permanent routing is done by nano-route. Special routing and nano routing are carried out with different metal layers.

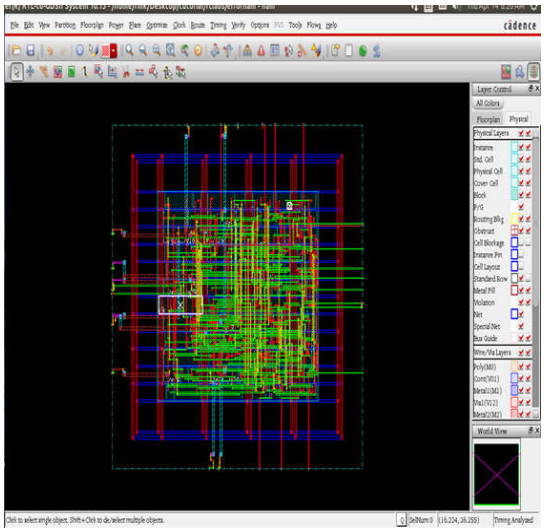


Figure 14. Physical view of Hamming coder

Fig.14 represents the layout view of the Hamming code.

Conclusion

On the comparison of various error detection and correction techniques, parity is the simplest form of error checking. CRCs seek to improve on checksums by increasing the complexity of the arithmetic. If the number of available pattern will get increased by adding the checksum, then the single-bit errors and double bit errors can be detected. Hamming codes are the extended idea of parity to include an error correction as well as an error-detection scheme. The essential idea in a Hamming code system is that a unique number is generated by parity errors which uniquely identifies the bit in error. Since bits can only take two values, the correct value is then known. On the overall analysis, CRC is simple to implement in binary hardware and mathematical analysis. It is good at detecting common errors caused by noise in transmission. The implemented model can be used in a digital data communication system to avoid distortions caused by the environment.

Acknowledgement

Thanks to department of CEME, AASTU for extending library support and to validate our idea successfully.

Conflict of Interest

The authors declared that there is no conflict of interst regading to this paper**.**

References

Chavan S., Sameer N. and Arunkumar P (2014). ASIC Implementation of High Throughput PID Controller. *Inter. J. Engi. Dev.Res.2(3): 3055-3060.*

Guirong W.U., Song J., Yuan W. and Ganggang Z. (2009). An Efficient Clock Tree Synthesis Method in Physical Design. Proceedings of the IEE International Conference of lectron Devices and Solid-State Circuits (EDSSC)., 190-193.

Martin G. and Steve B.F. (October 2011) " A Novel Programmable Parallel CRC Circuit," IEEE Trans. *Very Large Scale Integra. Syst., 19(10):1898-1902*

Ravi H. and Ashwini S.K. (2014). Design and Implementation of Hamming Code on FPGA using Verilog. *Inter.J.Engi. Adva.Techno., 4(2):180-184.*

Thornton M.A. (1997). Signed binary addition circuitry with inherent even parity out-puts. *IEEE Trans. Comput., 46(7): 811-816.*

Vishal B. and Amit U. (2014). Design of Multidirectional Parity Code Using Hamming Code Technique for Error Detection and Correction. *Paripex- Indian J. Res.3(5):79-81.*