Implementation of Low Power Wireless Sensor Node with Fault Tolerance Mechanism

R.Divyasharon[#1,] Dr.D.Sridharan[#2]
#1 Department of ECE, Anna University CEG, Chennai-25, India
#2 Department of ECE, Anna University CEG, Chennai-25, India

Abstract:

Wireless Sensor Network based solutions have been widely used since it has the potential to increase our ability to monitor and interact with our environment. WSN consist of number of wireless sensor nodes. Therefore, a node should have long time operating capability and efficient energy management. WSN are usually affected by noise which degrades the performance. It may be affected either by random error or burst error. In this paper, a low power sensor node with fault tolerance capability is developed using verilog code. Hamming code and Cyclic Redundancy Check is used for error free communication. The main aim of this project is to implement the design in FPGA because of its reprogramming ability and the power efficiency can be improved significantly in comparison with commercially micro-controller based sensor.

Keywords —*Low power sensor node, verilog, hamming code, cyclic redundancy check, VLSI, FPGA.*

I. INTRODUCTION

Wireless Sensor Network consists of numerous tiny sensor nodes deployed in regular or random manner. Sensor nodes must be low power, cheap and long lasting. WSN is used in various areas such as military applications, forest fire detection, flood detection, drug administrations in hospitals, home automation, vehicle tracking and detection and agriculture monitoring system [11]. Each sensor node consists of sensing, processing and transmission units. The sensing unit senses various ambient conditions including temperature, sound, light and motion around its location. The processing unit processes the data and splits it into packets. The transmission unit sends the packets to other higher order nodes. It is usually assumed that the transmission unit consumes more energy for data receiving and data sending. The Sensing unit consumes less power compared to the transmission unit. The processing unit, on the other hand consumes very less power compared to sensing and transmission unit. Therefore, it is important to put sensing and transmission unit into sleep mode whenever possible. Sleep scheduling can be used for energy management. In WSN, a wake up timer is used to activate the sensor node. When it receives a signal it gets activated and after processing it enters into sleep mode [2]. During transmission, the data may be affected by noise and other disturbances [10]. It degrades the performance of the network. These disturbances can change the shape and timing of the signal. If the data is binary encoded, they can alter the meaning of the data. These errors can be classified into two types. They are single-bit error and burst error. In single-bit error only one bit of data is changed from 1to 0 or vice-versa. It occurs least in serial data transmission and more in parallel data transmission. Burst error means that two or more bits in the data are changed from 1 to 0 or 0 to 1. Burst errors occur more in serial transmission.

For error detection and correction, the basic approach is to use redundant bits. The techniques mostly used are parity check, check sum and CRC. Hamming code is a linear error correcting code. It can detect up to two simultaneous bit errors and can correct one bit error [7]. CRC provides efficient protection against burst errors. It can detect all one bit error, two bit errors and all odd number of bits in error [9].

This paper focuses on implementation of low power sensor node with error detection and correction in FPGA. Because VLSI implementation is the most preferred solution for energy efficiency where after sensor deployment need for flexibility is almost nil.

II. RELATED WORKS

Liang et. al. designed a low power sensor node using wake up radio with frequency about 125 kHZ. Renyan Zhou et. al. designed a sensor node using 8051 whose clock frequency is 16 MHZ and its power dissipation is 80 uW [3].
Roshanzadeh et. al. used residue number system for error detection and correction in sensor node. In this paper, the technique to reduce traffic rates which decreases the amount of data transmission is presented. Error correction and detection using RNS with minimum redundancy is presented.

Vuran et. al. presented a cross-layer analysis of cross-layer analysis of error control schemes. This analysis compares automatic Repeat Request, Forward Error Correction and hybrid ARQ.

Wagner et. al. presented a simple lossless data compression algorithm for commercial sensor node using Huffman variable length code.

Pothare et. al. designed a Hamming code for 11 bit information using VHDL. Even parity is used for single bit error detection and correction [7].

Gupta et. al. designed a hamming code for 25 bit information using VHDL. Even and odd parity check method is used in this paper [8].

Campobello et. al. realized a high speed hardware for parallel CRC. CRC encoder and decoder is designed using recursive formula. Bergmann et. al. implanted CRC-16 in FPGA which can correct single bit error. Parallel implementation of CRC is used [9].

III. DESIGN AND IMPLEMENTATION

A. *Low Power Sensor Node*

Design of sensor node for a power efficient WSN faces some challenges. They are requirement of

- Large memory
- High computational ability
- Low power consumption
- Large bandwidth

Important design metrics for development of low power sensor node:

- To develop simple protocols to reduce power consumption
- Optimized design to reduce hardware involved
- To reduce the complexity of the circuit
- Minimum data storage

Figure 1 represents the operational flowchart for the sensor node in WSN [1]. It consists of two types of nodes. They are Coordinator node and sensor node. Coordinator node is the centralized controller. It generates a data packet. When the data packet is ready, it sends a signal to all other neighbor nodes to check whether they respond or not. If the coordinator node receives an acknowledgment, it transmits the data packet to the corresponding node. It owns a GPS system. It is a specialized sensor network with high processing capability.

Each sensor node contains LUT. LUT contains route information. If the path is modified, it requests the coordinator for the new route.

Figure 3 represents a block diagram of a sensor network. Coordinator node generates a data packet with the help of LUT and counter. It sends the data packet to the sensor node. Sensor node consists of processor and LUT. Sensor node processes the data. It contains LUT for route information. It sends the data packet to the next sensor node in the route.

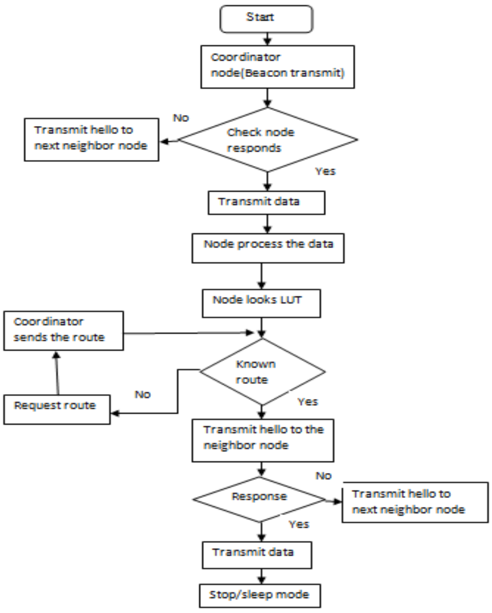

Start
Coordinator node(Beacon transmit)
Transmit hello to next neighbor node
No
Check node responds
Yes
Transmit data
Node process the data
Node looks LUT
Coordinator sends the route
Known route
Request route
No
Yes
Transmit hello to the neighbor node
No
Response
Transmit hello to next neighbor node
Yes
Transmit data
Stop/sleep mode

Fig. 1 The Operational Flowchart for Sensor Node in WSN

The format of the data packet is

Source address	Request ID	Destination address	Data packet	End signal

Fig. 2 Data Packet Form

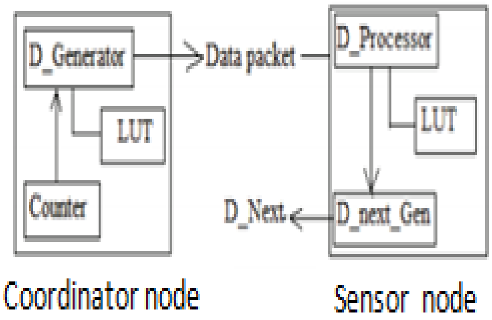
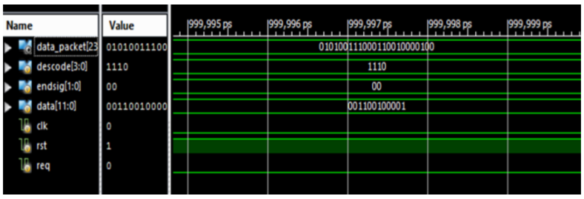


Fig. 3 Block Diagram of a Sensor Network

Low power sensor node has been designed and implemented in Xilinx ISE 14.2. Verilog coding is used to design the sensor node. The power, area and delay report has been generated using Cadence tool.



Name	Value
data_packet[23	01010011100
descode[3:0]	1110
endsig[1:0]	00
data[11:0]	00110010000
clk	0
rst	1
req	0

Fig. 4 Test bench Simulation of Coordinator Node

Figure 4 represents the test bench simulation of coordinator. Coordinator node generates a data packet of 24 bits which consist of 4 bit source address, 2 bit request id, 4bit destination address, 12 bit data packet and 2 bit end signal. Figure 5 represents the RTL schematic of a coordinator node.

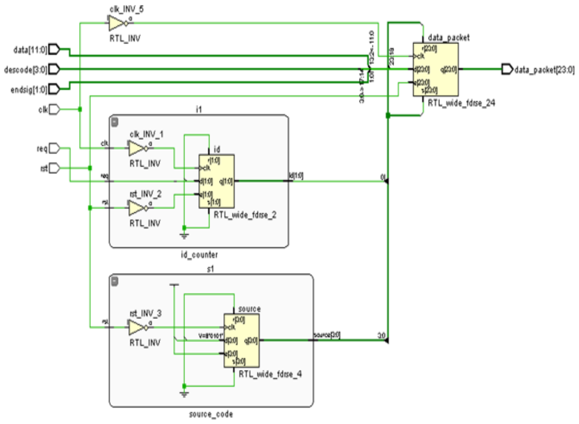


Fig. 5 RTL Schematic of cOordinator Node

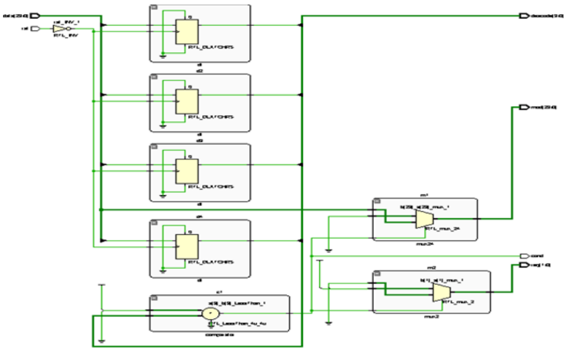
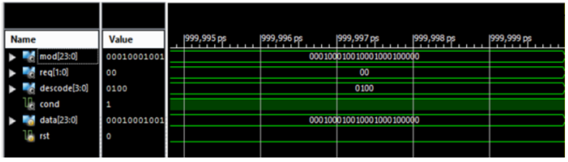


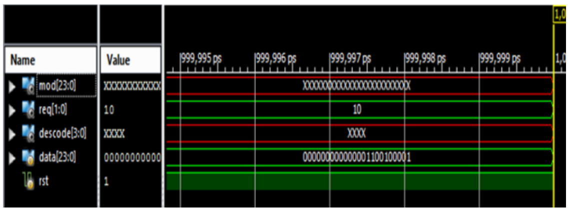
Fig. 6 RTL Schematic of Sensor Node

Figure 6 represents RTL schematic of a sensor node. Sensor node processes the data and sends it to next node in the route. It consists of D-latches, MUX and comparator.



Name	Value	
mod[23:0]	00010001001	000100010010001000100000
req[1:0]	00	00
descode[3:0]	0100	0100
cond	1	
data[23:0]	00010001001	000100010010001000100000
rst	0	

Fig. 7a known Destination



Name	Value	
mod[23:0]	XXXXXXXXXXXX	XXXXXXXXXXXXXXXXXXXXXXXX
req[1:0]	10	10
descode[3:0]	XXXX	XXXX
data[23:0]	00000000000	000000000000001100100001
rst	1	

Fig. 7b Unknown Destination

Fig. 7 Test bench Simulation of Sensor Node

Figure 7 represents the test bench simulation of sensor node. Figure 7a represents the known transmission. If the destination is known the node transmits the data to the next node in the route with request '00'. In fig 7b, if the destination is unknown the node sends request '01' to the coordinator node.

B. *Error Detection and Correction Using Hamming Code*

Richard Hamming invented hamming code. It can detect up to two bit errors and correct one bit error. This is done with the help of redundant bits. Codeword is formed by adding redundant bits with the data bits.

$$l=m+n$$

where l represents the size of code word, m represents the size of data bits and n represents the size of redundant bits.

Size of n is calculated by

$$2^n \geq m+n+1$$

Redundant bits are calculated from the data bits and its positions in the code word are 2^0, 2^1, 2^2…..$2^{(n-1)}$. It can be calculated by odd or even parity check method. Redundant bit values are calculated by using XOR gates. If the number of 1's in information bits is even, it produces the output 0 else 1.

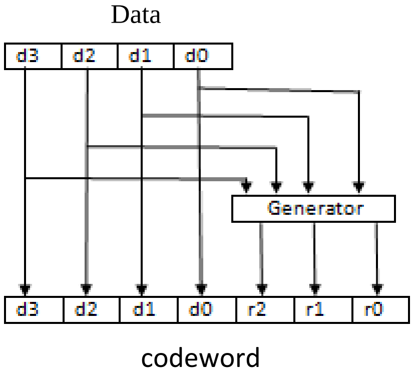
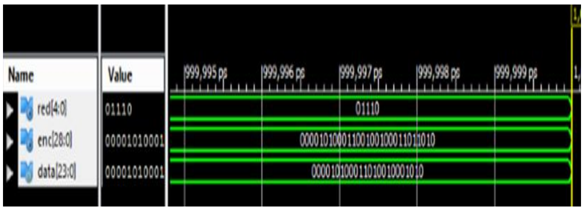


Fig. 8 Encoder Block Diagram

Figure 8 represents an encoder block diagram. Here information of 4 bit is used. Generator produces redundancy

bits of size 3. Codeword is formed by adding information bits and redundancy bits. This can be extended to any number of information bits.



Name	Value	
red[4:0]	01110	01110
enc[28:0]	00001010001	00001010001100100100011011010
data[23:0]	00001010001	000010100011010010001010

Fig. 9 Test Bench Simulation of Encoder.

In figure 9, information is of 16 bits. The redundancy bit of size 5 is generated. Codeword is generated by adding data and redundant bits.

Decoder block diagram consist of checker circuit which produces redundancy bits from the code word. Both redundancy bits are XORed to find the position of error and the error is corrected in the data.

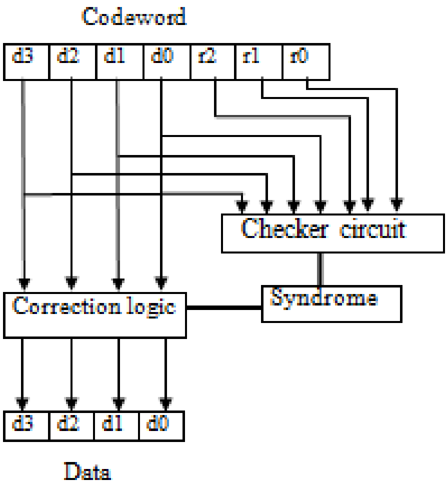
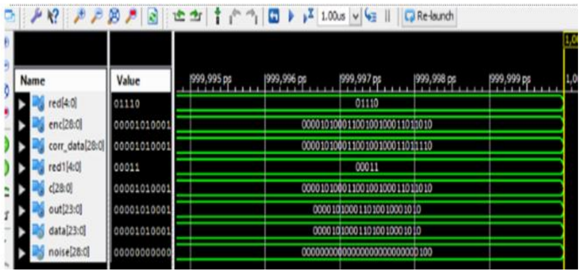


Fig. 10 Decoder Block Diagram



```
Name              Value          999,995 ps  999,996 ps  999,997 ps  999,998 ps  999,999 ps
red[4:0]          01110          01110
enc[28:0]         00001010001    00001010001100100100011011010
corr_data[28:0]   00001010001    00001010001100100100011011110
red1[4:0]         00011          00011
c[28:0]           00001010001    00001010001100100100011011010
out[23:0]         00001010001    000010100011010010001010
data[23:0]        00001010001    000010100011010010001010
noise[28:0]       00000000000    00000000000000000000000000100
```

Fig. 11 Test Bench Simulation of Decoder.

Here, corr_data represents the corrupted data. red1 represents the position of error. It denotes bit 3 is error. Hence 3rd bit is toggled and redundancy bits are removed to get the information bits[7,8].

C. *Error Detection and Correction Using Cyclic Redundancy Check*

CRC provides efficient protection against burst errors which commonly occurs in wireless transmission. CRC can detect all one bit, two bit and odd number of error bits. In this paper CRC-16 is implemented with sensor node which can detect multiple bit errors and correct single bit error. The polynomial used is $X^{16}+X^{12}+X^{5}+1$ in X25 standard. Parallel implementation of CRC is used here. This implementation involves two levels of logic and hence it is fast. Checksum equations are calculated by shifting and XORing. For 16-bit message, 16-bit checksum is added with the information to form a codeword. At the transmitter, $T=D_T\&C_T$, where T is the transmitted frame, D_T is the transmitted data and C_T is the transmitted checksum. At the receiver, $R=D_R\&C_R$, where R is the received frame, D_R is the received data and C_T is the received checksum. Checksum C_C is calculated at the receiver using same logic as that of transmitter. Both the checksum C_C and C_T is XORed. XOR pattern for all possible single bit error is calculated and stored in LUT. $C_{XOR}=C_T$ XOR C_C is calculated. If the result is zero, the information is error free else the result is compared with the values in LUT and the position of error bit is found and the corresponding bit is toggled at the receiver side[9].

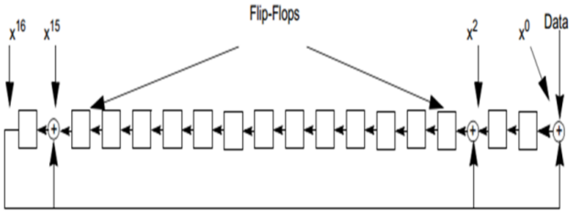
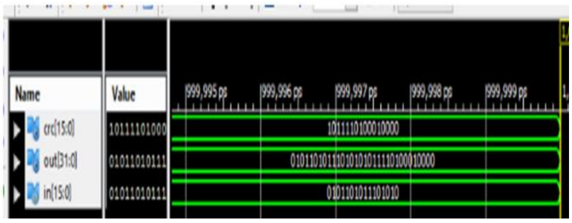


Fig. 12 CRC Block Diagram

The equations to find the CRC is generated by shifting and XORing as in figure 12.

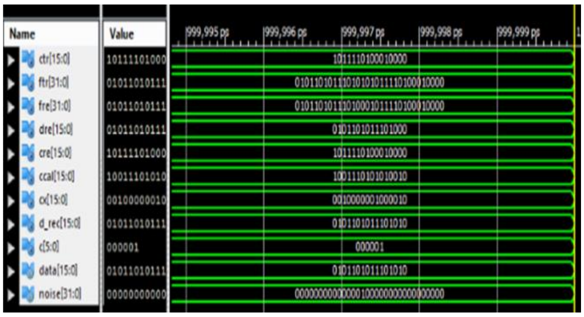
Figure 13 represents the test simulation of encoder, where CRC is generated from the equations computed. For 16 bit data, 16 bit checksum is generated and hence a codeword of 32 bit is generated.



```
Name        Value          999,995 ps  999,996 ps  999,997 ps  999,998 ps  999,999 ps
crc[15:0]   10111101000    1011110100010000
out[31:0]   01011010111    01011010111010101011110100010000
in[15:0]    01011010111    0101101011101010
```

Fig. 13 Test Bench Simulation Of Encoder.

CRC is calculated at the receiver side by using the equations computed before. Both CRC's are XORed to get the position of error. With the help of LUT corresponding bit is toggled to get error free information.



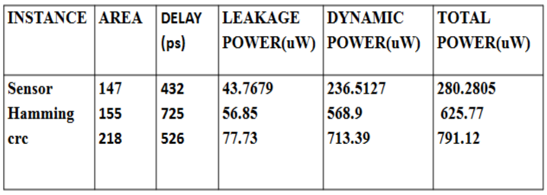
```
Name          Value          999,995 ps  999,996 ps  999,997 ps  999,998 ps  999,999 ps
ctr[15:0]     10111101000    1011110100010000
ftr[31:0]     01011010111    01011010111010101011110100010000
fre[31:0]     01011010111    01011010111010001011110100010000
dre[15:0]     01011010111    0101101011101000
cre[15:0]     10111101000    1011110100010000
ccal[15:0]    10011101010    1001110101010010
cx[15:0]      00100000010    0010000001000010
d_rec[15:0]   01011010111    0101101011101010
c[5:0]        000001         000001
data[15:0]    01011010111    0101101011101010
noise[31:0]   00000000000    00000000000000100000000000000000
```

Fig. 14 Test Bench Simulation of Decoder.

IV. PERFORMANCE

Power, area and delay of sensor node, CRC, hamming is estimated using cadence tool.

Table I : Power, Area and Delay Report



INSTANCE	AREA	DELAY (ps)	LEAKAGE POWER(uW)	DYNAMIC POWER(uW)	TOTAL POWER(uW)
Sensor	147	432	43.7679	236.5127	280.2805
Hamming	155	725	56.85	568.9	625.77
crc	218	526	77.73	713.39	791.12

Area and power consumption of hamming code is less compared to cyclic redundancy check. But delay of hamming code is slightly higher than CRC. Wireless communication is usually affected by burst error. Hence, it is better to use CRC for error detection and correction.

V. CONCLUSION

Low power sensor node with fault tolerance mechanism is designed and implemented in Xilinx ISE 14.2 using verilog. The power dissipation, area and delay have been estimated for low power sensor node and error detection and correction mechanisms using cadence. Power consumption of sensor node is reduced in terms of uW in VLSI implementation. Low power sensor node with error detection and correction presented in this paper gives better performance compared with micro-controller based sensors. The other major advantage in VLSI implementation is its reprogramming capability.

ACKNOWLEDGMENT

This shall not be completed without the benefaction of the Almighty God. We would like to express our profound gratitude towards him. We would like to express our special gratitude to our parents and friends who greatly accompanied us both with financial and academic support.

REFERENCES

[1] J.Bag, S.Roy, S.K.Sarkar, "Realization of a low power sensor node processor for Wireless Sensor Network and its VLSI implementation" IEEE international Advance computing conference, 2014,pp.101-105.

[2] F. Akyildiz, W. Su, Y. Sandarasurbramaniam, E. Cayirci, "Wireless sensor networks: a survey" Computer Networks Journal, Elsevier vol. 38 no. 4. pp. 393-422. 2002.

[3] Renyan Zhou, Leibo Liu, Shouyi Yin, Ao Luo, Xinkai Chen, Shaojun Wei, " A VLSI Design of Sensor Node for Wireless Image", IEEE, pp. 149-152. 2010

[4] Aki Hopponen, "Low power design for Wireless Sensor Networks" Springer, 2012.

[5] G. Lu, N. Sadagopan, B. Krishnamachari and A. Goel. "Delay efficient sleep scheduling in Wireless sensor networks" in Proc. of IEEE INFOCOM, 2005, vol. 4, pp 24702481.

[6] G. U. Gamm, M, Kostic, M. Sippel and L. M. Reindl, "Low-power sensor node with addressable wake-up on-demand capability" Int. Journal of Sensor Networks, Inderscience, Vol. ll, No. l, pp. 49-56. 2012

[7] Pranjali Pothare, Prajakta Ambatkar, "Hamming code for single bit error detection & error correction with even parity using VHDL" International Journal of Advanced Research in Computer Engineering & Technology, Volume 4 Issue 1, January 2015

[8] Rajesh Kumar Gupta , Prof. Rajeshwar Lal Dua , "30 BIT Hamming Code for Error Detection and Correction with Even Parity and Odd Parity Check Method by using VHDL" International Journal of Computer Applications (0975 – 8887),Volume 35– No.13, December 2011.

[9] S. Shukla , N. W. Bergmann , "Single bit error correction implementation in CRC-16 on FPGA"IEEE conference on field-programmable technology, 2014,pp.319-322.

[10] B.Sakthivel, P.Nagarajan, B.Sivananthan, "Detection and prevention of congestion attacks Using Multivariate Correlation Analysis", SSRG International Journal of Computer Science and Engineering (SSRG-IJCSE), volume1 issue9, Nov 2014.

[11] Sirisha D, B Venkateswaramma, M Srikanth and A Anil Babu , "Wireless Sensor Based Remote Controlled Agriculture Monitoring System Using ZigBee" , SSRG International Journal of Electronics and Communication Engineering (SSRG-IJECE),volume 2 Issue 4 April 2015.

[12] Harshada L. Borkar, V.N. Bhonge, "Design And Implementation Of Reed Solomon Encoder And Decoder",SSRG International Journal of Electronics and Communication Engineering (SSRG-IJECE) , volume 2 issue 1 Jan 2015