Implementation of Energy Efficient LDPC Code for Wireless Sensor Node

Sang-Min Choi[1] and Byung-Hyun Moon[2]

[1] D&D Division
Taihan Electric Wire Co., Ltd.
785, Gwangyang-2dong, Dong-gu, Anyang, Gyeonggi, 431-810, Korea
smchoi80@taihan.com
[2] Dept. of Computer and Communication Engineering, Daegu Univ.
15, Neari, Jinryang, Gyeongsan, Gyeongbuk, 712-714, Korea
bhmoon@daegu.ac.kr

Abstract. The energy efficiency of error control scheme is very important because of the strict energy constraints of wireless sensor networks. Wireless sensor node requires simple error control schemes because of the low complexity request of sensor nodes. Automatic repeat request(ARQ) and forward error correction(FEC) are the key error control strategies in wireless sensor networks. In this paper, we implemented the efficient QC-LDPC encoder which does not require matrix inversion to improve the complexity of the encoder. It is shown that the efficient QC-LDPC code obtained 17.9% and 36% gain respectively in the mean number of transmission for the transmission power of -19.2dBm and -25dBm.

Keywords: QC-LDPC code, Wireless sensor networks.

1 Introduction

In recent years, the idea of wireless sensor networks has produced lots of research, because of wireless sensor networks can be applied widely in many fields. In wireless sensor networks, erroneous transmission can happen by wireless channel noise. Automatic repeat request (ARQ) detects errors using cyclic redundancy check (CRC) and retransmission of data is used as error control scheme for wireless sensor networks. Sensor node requires long lifetime with limited battery, but retransmission of data is the primary source of energy consumption and reduces the lifetime of sensor node. Therefore, error control scheme of forward error correction (FEC) for wireless sensor networks is necessary [1].

Low-density parity-check (LDPC) codes were first introduced by Gallager in 1962 and rediscovered by Mackay and Neal in 1996 and come into the spotlight for next generation communication system[2][3]. The LDPC codes construction can be categorized into Mackay's random construction and sub-block based structured construction. The Mackay's random construction LDPC codes show very good performance. However, this construction is computationally intensive implementation because of large memory requirement. The sub-block based structured construction

T.-h. Kim et al. (Eds.): FGCN 2011, Part II, CCIS 266, pp. 248–257, 2011.

scheme can be implemented with less complexity than the Mackay's random construction [4].

In this paper, the efficient quasi-cyclic(QC) LDPC code on ATmega128 based sensor node using sensor network OS platform(SenWeaver OS) is implemented. The efficient QC-LDPC code for wireless sensor networks has small size parity check matrix and can be implemented easily. The encoding scheme of efficient QC-LDPC code is simplified version of Richardson's encoder and does not require matrix inversion so that this encoding scheme is suitable for sensor node which has limited computation ability. This paper is organized as follows. In section 2, the efficient QC-LDPC encode is given. In section 3, the specifications of the sensor node and the OS platform (SenWeaver OS) are introduced. In section 4, the experiment results for the efficient QC-LDPC code are given. Finally, the conclusion is made in section 5.

2 The Efficient QC-LDPC Code

2.1 QC-LDPC Code

For wireless sensor node applications, we consider a subclass of LDPC code, QC-LDPC, whose parity-check matrix consists of circulant permutation matrices or the zero matrix. I_i is the $N_s \times N_s$ permutation matrix which shifts the identity matrix I to the right by i-times for any integer i, $0 \le i \le N_s$. Let I_1 be the $N_s \times N_s$ permutation matrix given by

$$I_1 = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \end{bmatrix} \tag{1}$$

Using this notation parity check matrix H can be defined by

$$H = \begin{bmatrix} I_{s_{0,0}} & I_{s_{0,1}} & I_{s_{0,2}} & \cdots & I_{s_{0,(n-1)}} \\ I_{s_{1,0}} & I_{s_{1,1}} & I_{s_{1,2}} & \cdots & I_{s_{1,(n-1)}} \\ I_{s_{2,0}} & I_{s_{2,1}} & I_{s_{2,2}} & \cdots & I_{s_{2,(n-1)}} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ I_{s_{(m-1),0}} & I_{s_{(m-1),1}} & I_{s_{(m-1),2}} & \cdots & I_{s_{(m-1),(n-1)}} \end{bmatrix} \tag{2}$$

where $s_{i,j}$ is the shift value corresponding position (i,j) sub-block. This value is one of the {0, 1, 2, …, N_s-1} for nonzero sub-block and $s_{i,j} = -1$ for zero matrix. The size of H is $mN_s \times nN_s$. [4].

2.2 Encoding of LDPC Codes

Block type LDPC encoding process has Richardson's encoding algorithm if $M \times N$ Parity check matrix is divided into the form

$$H = \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix} \tag{3}$$

where A is $(M\text{-}l) \times (N\text{-}M)$, B is $(M\text{-}l) \times l$, T is $(M\text{-}l) \times (M\text{-}l)$, C is $l \times (N\text{-}M)$, D is $l \times l$, E is $l \times (M\text{-}l)$. All these matrices are sparse and T is a lower triangular with one along the diagonal [5]. Fig. 1 shows the parity check matrix for the Richardson's encoder.

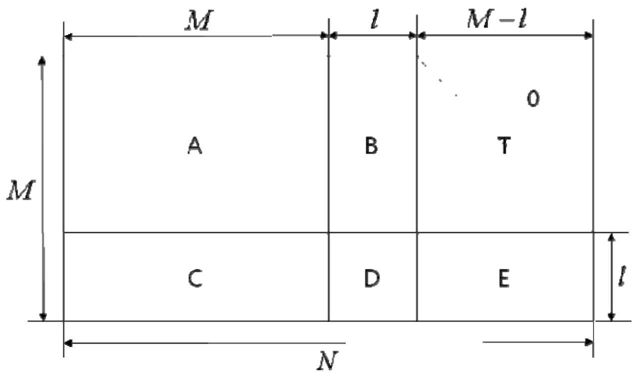

Fig. 1. Parity check matrix for Richardson's encoder

Let the codeword c={u, p_1, p_2} where u denotes the systematic part, p_1 and p_2 denote the parity parts, p_1 has length l, and p_2 has length $(M\text{-}l)$. The codeword c satisfies the following equations.

$$Hc^T = \begin{bmatrix} A & B & T \\ C & D & E \end{bmatrix} \begin{bmatrix} u^T \\ p_1^T \\ p_2^T \end{bmatrix} = 0^T \tag{4}$$

$$\begin{aligned} &Au^T + Bp_1^T + Tp_2^T = 0 \\ &\left(-ET^{-1}A + C\right)u^T + \left(-ET^{-1}B + D\right)p_1^T = 0 \end{aligned} \tag{5}$$

Let $\Phi = -ET^{-1}B + D$ and assume that Φ is nonsingular. Then, we can obtain parity bits as the following equations

$$\begin{aligned} p_1^T &= -\Phi^{-1}\left(-ET^{-1}A + C\right)u^T \\ p_2^T &= -T^{-1}\left(Au^T + Bp_1^T\right) \end{aligned} \tag{6}$$

2.3 The Efficient QC-LDPC Code

The encoding scheme of efficient QC-LDPC code is to simplify the Richardson's encoder. The goal is to eliminate the matrix inversion of Φ^{-1} and T^{-1} in equation (6). If Φ and T are identity matrixes, the encoder can be simplified as shown in equation (7). Also, the simplified encoder does not require matrix inversion and encoding can be done without sub-matrix D and T. This will reduce amount of computations required to encode at wireless sensor nodes. Fig. 2 shows the simplified Richardson's encoder.

$$\begin{aligned} p_1^T &= -\left(-EA + C\right)u^T \\ p_2^T &= -\left(Au^T + Bp_1^T\right) \end{aligned} \tag{7}$$

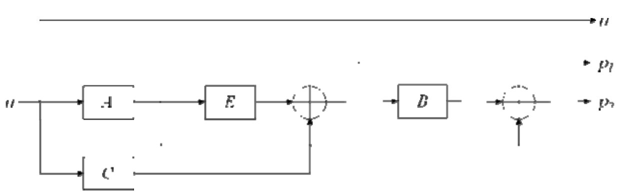

Fig. 2. The simplified Richardson's encoder

The parity check matrix for the simplified Richardson's encoder is constructed by 3×6 sub-matrices as shown in equation (8).

$$H = \left[\begin{array}{ccc|c|cc} A_{1,1} & A_{1,2} & A_{1,3} & A_{1,4} & A_{1,5} & A_{1,6} \\ A_{2,1} & A_{2,2} & A_{2,3} & A_{2,4} & A_{2,5} & A_{2,6} \\ \hline A_{3,1} & A_{3,2} & A_{3,3} & A_{3,4} & A_{3,5} & A_{3,6} \end{array}\right] \tag{8}$$

In equation (8), to make T matrix as identity matrix, let $A_{1,5} = A_{2,6} = I$ and $A_{1,6} = A_{2,5} = I_{-1} = 0$. To make Φ as an identity matrix from equation (9), equation (10) has to be satisfied.

$$\begin{aligned}\Phi &= -ET^{-1}B - D \\ &= \begin{bmatrix} A_{3,5} & A_{3,6} \end{bmatrix} \begin{bmatrix} A_{1,4} \\ A_{2,4} \end{bmatrix} + A_{3,4} \\ &= A_{3,5}A_{1,4} + A_{3,6}A_{2,4} + A_{3,4} = I\end{aligned} \tag{9}$$

Also, if we let $A_{3,4} = I$ which corresponds to matrix D as identity matrix, equation (10) is obtained.

$$A_{3,5}A_{1,4} + A_{3,6}A_{2,4} = 0 \tag{10}$$

Assume $A_{3,5}A_{1,4} = A_{3,6}A_{2,4}$, then Φ becomes identity matrix. In order to satisfy equation (10), we let $A_{1,4}$ in B matrix and $A_{3,6}$ in E matrix as an identity matrix. And, let $A_{3,5}$ and $A_{2,4}$ are equivalent matrixes. The parity check matrix for the efficient QC-LDPC code is shown in Fig. 3.

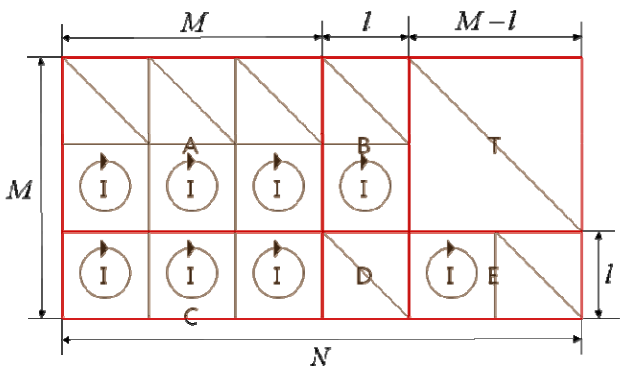

Fig. 3. Parity check matrix for the efficient QC-LDPC code

In this paper, we propose 96×192 parity check matrix that is made by shifting the 32×32 identity matrix as shown in the following equation (11).

$$H = \left[\begin{array}{ccc|c|cc} I & I & I & I & I & I_{-1} \\ I_1 & I_3 & I_5 & I_7 & I_{-1} & I \\ \hline I_{13} & I_{11} & I_9 & I & I_7 & I \end{array}\right] \tag{11}$$

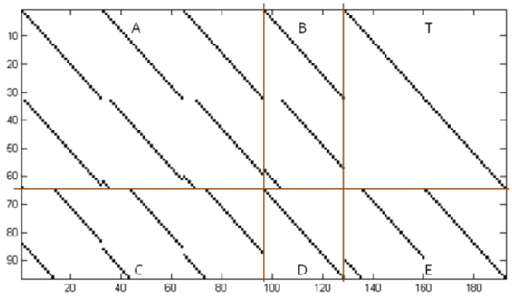

Fig. 4. Parity check matrix of 96×192 for the efficient QC-LDPC code

Code rate 1/2 Mackay's random constructed LDPC code with column weight 3 and code length 192 has six cycle-4. However the efficient QC-LDPC code has zero cycle-4 and girth of 6. The parity check matrix for the efficient 96×192 QC-LDPC is shown in Fig. 4

3 Specifications of Sensor Node and OS Flatform

We have implemented an energy efficient QC-LDPC code on an ATmega 128 based sensor node as shown in Fig. 5. The detailed specifications of the sensor node are shown in Table 1. The extended 32Kbyte SRAM by Chiplus is used to overcome the shortage of the memory during the encoding and decoding of QC-LDPC. The CC2420 RF module is used to transmit the LDPC coded message. The multi-layer chip antenna for the frequency range between 2400-2455MHz with 100 MHz bandwidth is used.

The sensor node uses SenWeaver OS that is developed by the UTRC(Ubiquitous Technology Research Center) in Daegu University [6]. The features of SenWeaver OS are as following

- Priority based scheduling/multi-threading
- Vertical and horizontal layered architecture
- Provide ANSI C based API(Application Program Interface)
- Semantic modular architecture (energy efficiency by dynamic software reprogramming)
- Provide multiprogramming like PC that operate a number of API without alteration
- Provide hierarchical reconstruction of API
- Provide abstraction and automatic code generation of variable sensor node hardware.

The software architecture for sensor nodes designed such that a layer modular is a separate block. This allows easy replacement a block according to hardware without changing upper layers. The SenWeaver OS architecture overview is shown in Fig. 6.

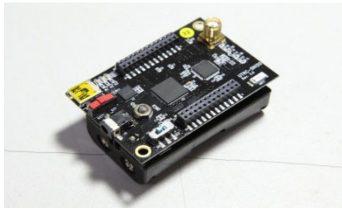

Fig. 5. Sensor node used for the efficient QC-LDPC code implementation

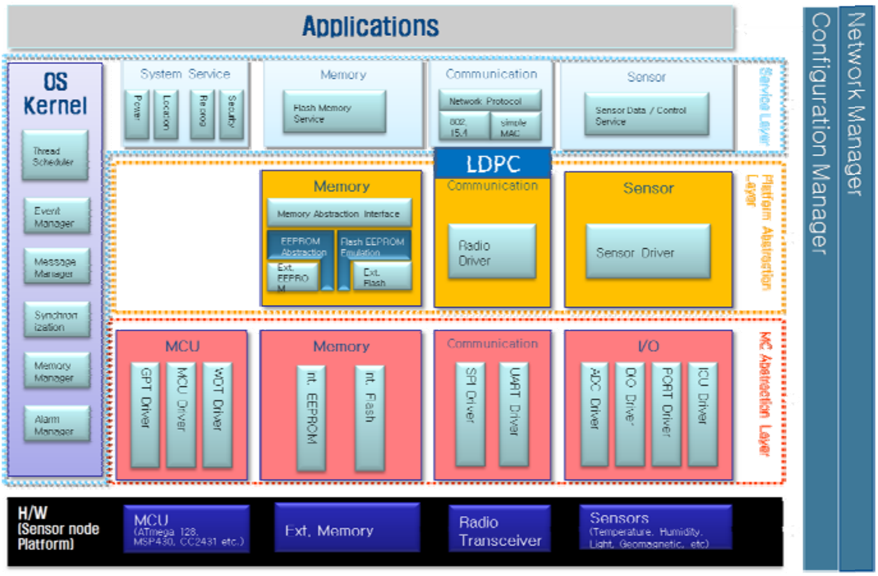

Fig. 6. SenWeaver OS architecture overview

Table 1. Detailed specifications of a sensor node

	Model	Specification
MCU	ATmega128L	8bit processor, 128Kbytes flash, 4Kbyte SRAM
Ext. SRAM	CS18LV02563	32Kbytes SRAM
RF module	CC2420	2.4GHz, IEEE 802.15.4/ZigBee-ready RF transceiver
Antenna	SWBBL1	Multilayer chip antenna 2400~2485MHz, 100MHz BW

4 Experiment Results

As shown in Fig. 6, the transmitter and the receiver are located on the ceiling to ensure the line of sight communication. By monitoring the 2.43GHz frequency band, no other source of interference is observed. In order to measure performance of the efficient QC-LDPC code, 15,000 12byte long MAC frames as shown in Fig. 7 are transmitted.

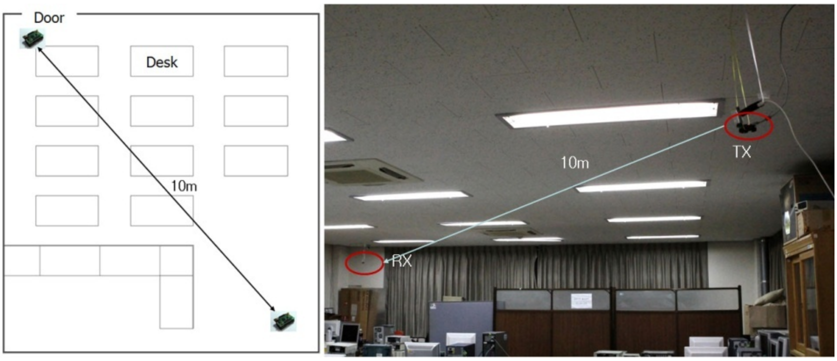

Fig. 7. Experiment environment of performance measurement

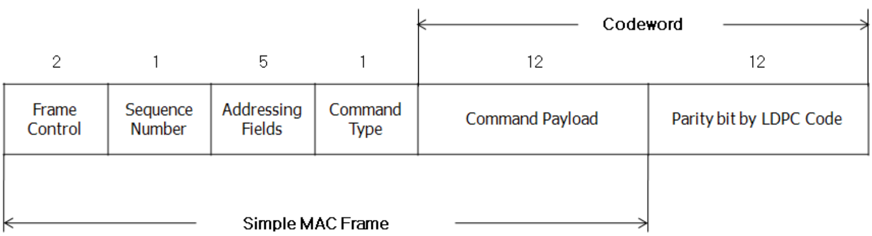

Fig. 8. LDPC coded simple MAC frame

The experiment results for the uncoded and the efficient QC-LDPC coded cases are summarized in Table 2. The transmitting power is varied from -7 dBm to -25 dBm..

Table 2. Measured BER performance of efficient QC-LDPC code

TX power	Uncoded case	Efficient QC-LDPC coded case
-7 dBm	3.4787×10^{-6}	0
-10 dBm	8.5301×10^{-5}	5.7418×10^{-5}
-15 dBm	8.3478×10^{-4}	4.6648×10^{-4}
-19.2 dBm	7.0767×10^{-3}	5.0378×10^{-3}
-25 dBm	1.6713×10^{-2}	1.2139×10^{-2}

Given the packet length and BER performance, we calculate the PER (Packet Error Rate) to be

$$P_{ep} = 1 - (1 - BER)^{L} \tag{12}$$

where P_{ep} is PER, L is packet length[7]. The calculated PER using equation (12) is shown in table 3.

Table 3. PER performance analysis of efficient QC-LDPC code

TX power	Uncoded case	Efficient QC-LDPC coded case
-7 dBm	3.3390×10^{-4}	0
-10 dBm	8.1558×10^{-3}	5.4971×10^{-3}
-15 dBm	7.7043×10^{-2}	4.3804×10^{-2}
-19.2 dBm	4.9428×10^{-1}	3.8421×10^{-1}
-25 dBm	8.0171×10^{-1}	6.9040×10^{-1}

The mean number of transmission, R, required for success can be calculated as

$$R = (1 - P_{ep}) \sum_{k=0}^{\infty} (k+1) P_{ep}^{k} = \frac{1}{1 - P_{ep}} \tag{13}$$

Also, the gain of mean number of transmission R_{gain} is given by

$$R_{gain} = \frac{R_{coded}}{R_{uncoded}} \tag{14}$$

Where R_{uncoded} is mean number of transmission for uncoded case, R_{coded} is that of the efficient QC-LDPC coded case. Table 4 shows the mean number of transmission using equations (13) and (14).

Table 4. Mean number of transmission for success

TX power	Uncoded case	Efficient QC-LDPC coded case	R_{gain}
-7 dBm	1.0003	1.0000	0.0003
-10 dBm	1.0082	1.0055	0.0027
-15 dBm	1.0835	1.0458	0.0348
-19.2 dBm	1.9774	1.6239	0.1787
-25 dBm	5.0430	3.2300	0.3595

It is shown that the implemented QC-LDPC code showed limited gain for the TX powers between -7dBm and -10dBm. When the TX powers are between -19.2dBm and -25dBm, the implemented QC-LDPC code obtained about17.9% and 36% gain over uncoded case, respectively.

5 Conclusion

In this paper, we propose FEC using efficient QC-LDPC code for the error control in the wireless sensor networks. The efficient QC-LDPC code has small size parity check matrix and easy implementation scheme. The encoding of the efficient QC-LDPC code that constructs sub-matrix T and Φ of Richardson's encoder into identity matrix and requires less computation compare to the Mackay's random constructed LDPC code.

We implemented the efficient QC-LDPC code on ATmega128 based sensor node using sensor network OS platform (SenWeaver OS) and measured the BER performance at TX powers between -7dBm to -25dBm. Also, we calculated the packet error rate and the mean number of transmission required for successful transmission.

It is shown that the implemented QC-LDPC code obtained performance gain in the mean number of transmission about 17.9% and, 36% gain over the uncoded case for the transmission powers of -19.dBm and -25dBm, respectively.

Acknowledgments. This research was financially supported by Daegu University Research Grant, 2011.

References

1. Jeong, J., Tien Ee, C.: Forward error correction in sensor Networks, U.C.Berkeley Technical Report (May 2003)
2. Gallager, R.G.: Low-Density parity-Check Codes. IRF Trans. on Info. Theory 8, 21–28 (1962)
3. MacKay, D.J.C., Neal, R.M.: Near Shannon limit performance of low density parity check codes. Electron, Lett. 32(18), 1645–1646 (1996)
4. Fossorier, M.P.C.: Quasi-Cyclic Low Density Parity Check Codes from Circular Permutation Matrices. IEEE Trans. Information Theory 50, 1788–1794 (2004)
5. Richardson, T.J., Urbanke, R.L.: Efficient Encoding of Low-Density Parity-Check Codes. IEEE Trans. IT 47, 638–656 (2001)
6. Kim, T.-H., Kim, H.-C.: Software Architecture for Highly Reconfigurable Sensor Operating System. Journal of IEMEK 2(4), 242–250 (2007)
7. Lettieri, P., Fragouli, C., Srivastava, M.: Low power error control for wireless links. In: Proceedings of 3rd annual ACM/IEEE Intl. conference on Mobile computing and networking (MOBICOM) (1997)