International Journal of Research
Available at https://edupediapublications.org/journals
e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 07
March 2018

Design of Single Error Correction codes with fast decoding of control bits

V. Sudharani & Smt.J.Sofia Priya Dharshini (M. tech, P.HD)
sudhasri473@gmail.com & jspd1810@gmail.com
[1]PG Scholar, VLSI, RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY, Nandyal, Kurnool, AP.
[2]Assistant Professor, Department of ECE, RAJEEV GANDHI MEMORIAL COLLEGE OF ENGINEERING & TECHNOLOGY, Nandyal, Kurnool, AP.

Abstract: *As the technology scales down, shrinking geometry and layout dimension, on-chip interconnects are exposed to different noise sources such as crosstalk coupling, supply voltage fluctuation and temperature variation that cause random and burst errors. Hence, error correction codes integrated with noise reduction techniques are incorporated to make the on-chip interconnects robust against errors. Single error correction (SEC) codes are widely used to protect data stored in memories and registers. In some applications, such as networking, a few control bits are added to the data to facilitate their processing. For example, flags to mark the start or the end of a packet are widely used. Therefore, it is important to have SEC codes that protect both the data and the associated control bits. It is attractive for these codes to provide fast decoding of the control bits, as these are used to determine the processing of the data and are commonly on the critical timing path. In this brief, a method to extend SEC codes to support a few additional control bits is presented.*

Index Terms: Error correction codes, high-speed networking, memory, single error correction (SEC).

I. INTRODUCTION

The repetition of a corrupt message is a problem in real time communications, where the data should be delivered with low delay, for which the use of techniques avoiding overloads by transmitting are adequate.

Once the devices are in the field, other reliability issues appear in the form of soft errors or age induced permanent failures. Memory devices are among those affected by those issues due to their high level of integration. Current techniques to address those reliability issues in memories include the use of redundant elements to repair manufacturing defects, and the use of Error Correcting Codes (ECC) to deal with soft errors once the device is in operation. Different techniques are used to deal with defects versus soft errors. ECC can also be used to correct errors caused by defects, but then their ability to correct soft errors may be compromised leading to a reduced reliability. However, to the best of our knowledge, there is no previous work on how the use of ECC to deal with defects affects the reliability of memory in the field. In this paper, an effective technique to use ECC to deal with isolated defects and soft errors on memory chips is presented.

NETWORKING applications require high-speed processing of data and thus rely on complex integrated circuits. In routers and switches, packets typically enter the device through one port, are processed, and are then sent to one or more output ports. During this processing, data are stored and moved through the device. Reliability is a key requirement for networking equipment such as core routers. Therefore, the stored data must be protected to detect and correct errors. This is commonly done using error-correcting codes (ECCs). For memories and registers, single error correction (SEC) codes that can correct 1-bit errors are commonly used.

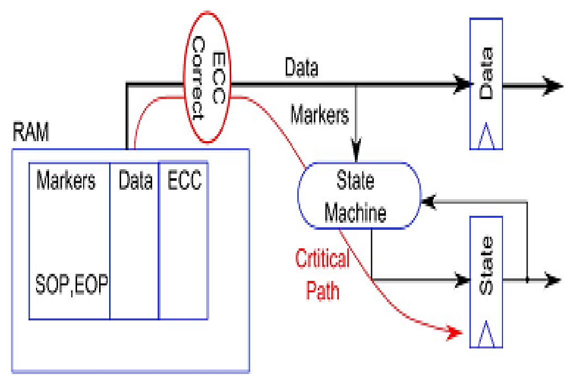


Fig.1 Typical packet data storage in a networking application

One option is to protect the data and the control bits as different data blocks using separate ECCs. For example, let us assume 128-bit data blocks with 3 control bits. Then, a SEC code can protect a data block using 8 parity check bits, and another SEC code can protect the 3 control bits using 3 parity check bits. In the resulting codes, the control bits can be decoded using a subset of the parity check bits. This reduces the decoding delay and makes them suitable for networking applications. To evaluate the method, several codes have been constructed and implemented. They are then compared with existing solutions in terms of decoding delay and area.

Background coding theory

Background coding theory more detailed accounts of error-correcting codes can be found in: Hill, Pless, MacWilliams and Sloane, van Lint, and Assmus and Key. See also Peterson for an early article written from the engineers' point of view. Proofs of all the results quoted here can be found in any of these texts; our summary here follows. The usual pictorial representation of the use of error-correcting codes to send messages over noisy channels is shown in the schematic diagram

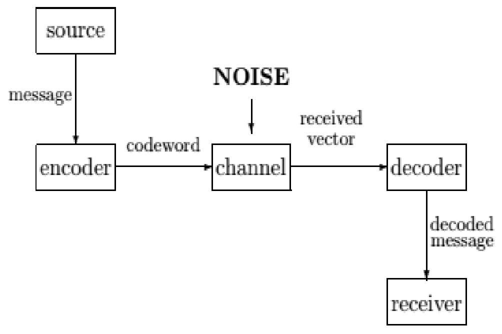
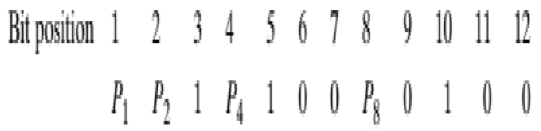


Figure 2: A noisy communications channel

Here a message is first given by the source to the encoder that turns the message into a codeword, i.e. a string of letters from some alphabet, chosen according to the code used.

Hamming codes

The most common types of error-correcting codes used in RAM are based on the codes devised by R. W. Hamming. In the Hamming code, k parity bits are added to an n-bit data word, forming a new word of n + k bits. The bit positions are numbered in sequence from 1 to n k. Those positions numbered with powers of two are reserved for the parity bits. The remaining bits are the data bits. The code can be used with words of any length. Before giving the general characteristics of the Hamming code, we will illustrate its operation with a data word of eight bits. Consider, for example, the 8-bit data word 11000100. We include four parity bits with this word and arrange the 12 bits as follows:



Bit position	1	2	3	4	5	6	7	8	9	10	11	12
	P_1	P_2	1	P_4	1	0	0	P_8	0	1	0	0

The 4 parity bits P1 through P8 are in positions 1, 2, 4,and 8, respectively. The 8 bits of the data word are in the remaining positions. Each parity bit is calculated as follows:

$$P_1 = \text{XOR of bits } (3,5,7,9,11) = 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 = 0$$

$$P_2 = \text{XOR of bits } (3,6,7,10,11) = 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 = 0$$

$$P_4 = \text{XOR of bits } (5,6,7,12) = 1 \oplus 0 \oplus 0 \oplus 0 = 1$$

$$P_8 = \text{XOR of bits } (9,10,11,12) = 0 \oplus 1 \oplus 0 \oplus 0 = 1$$

Recall that the exclusive-OR operation performs the odd function. It is equal to 1 for an odd number of 1's among the variables and to 0 for an even number of 1's. Thus, each parity bit is set so that the total number of 1's in the checked positions, including the parity bit, is always even.

Data Protection In Networking Applications.

Modern networking equipment supports data rates that range from 10 to 400 Gbit/s, and terabit rates are expected in the near future. The clock frequencies used in current ASICs are typically in the range of 300 MHz to 1 GHz, and the clock frequencies in FPGAs are typically lower (under 400 MHz). To support these high data rates, on-chip packet data buses are wide, with typical widths between 64 and 2048 bits.

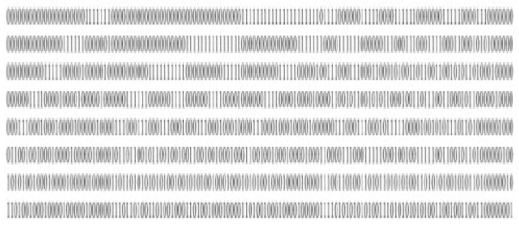


Fig.3. Parity check matrix for a minimum-weight SEC code that protects 128 data bits.

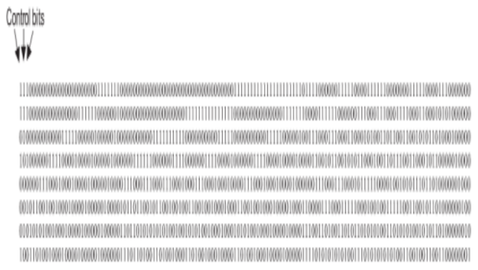

Control bits

Fig 4. Parity check matrix for a minimum-weight SEC code that protects 128 data and 3 control bits.

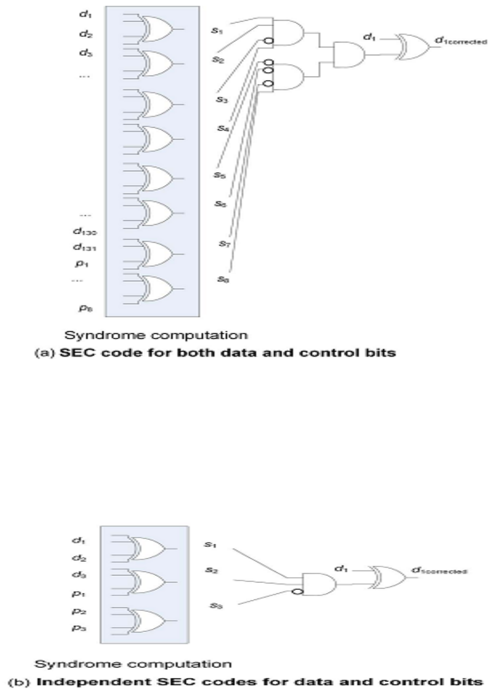


Fig.5. Decoding of a control bit for single and independent SEC codes for data and control. (a) SEC code for both data and control bits. (b) Independent SEC codes for data and control bits.

Packet data must frequently be stored in RAMs, e.g., in FIFOs for adapting processing rates. When storing packet data, it is necessary to delineate the packet boundaries. In the absolute simplest case, each segment on the bus can be delineated with a single EOP marker. The next valid segment is then assumed to be the start of the following packet. where a packet is in error and it must be dropped. To mark such eroded packets, an additional control signal (ERR) may be required.

II. LITERATURE SURVEY

Using Single Error Correction Codes to Protect Against Isolated Defects and Soft Errors

The technology scaling process provides high-density, low cost, high-performance integrated circuits. These circuits are characterized by high operating frequencies, low voltage levels, and small noise margins with increased defect rate. To cope with defects in memory chips, many different techniques have been proposed, all of them based on the use of redundant elements to replace defective ones. Those techniques vary from those applied during the manufacturing process, in the test phase, to the use of built-in circuits able to repair the memory chips even during normal operation in the field, with different tradeoffs in terms of cost and speed. The use of redundant rows and columns has been widely used in memory design to cope with this problem. One-dimensional (1-D) redundancy is the simplest variation in which only redundant rows (or columns) are included in the memory array and used to replace the defective rows (or columns) detected during test. The main advantage of this approach is that its implementation does not require any complex allocation algorithms. Unfortunately, its repair efficiency can be low because a defective column (row) containing multiple defective cells cannot be replaced by a single redundant row (column). Examples of such techniques are presented. Authors proposed a two-dimensional (2-D) redundancy approach which improves the efficiency of the 1-D approach. This approach adds both redundant rows and columns to the memory array to provide more efficient repair when multiple defective cells exist in the same row or column of the array. When multiple faulty cells are detected, the choice between the use of a redundant row or a redundant column to replace them is made based on the maximum repair capability of each alternative.

The main drawback of this approach is that the optimal redundancy allocation problem. Although many heuristic algorithms have been proposed to solve this problem, it is still difficult to develop built-in repair implementations using them. For both redundancy approaches, when the number of defective cells in the array exceeds the repair capability through the use of redundant elements, the last alternative before discarding the defective chip is to try to use it as a downgraded version of memory

Multi bit random and burst error correction code with crosstalk avoidance for reliable on chip interconnection links

To increase the performance of the NoC interconnect, many research groups proposed FEC coding and joint error correction coding with crosstalk avoidance. Single error correcting (SEC) Hamming code, single error correction and double error detection (SEC–DED) extended Hamming code is used to correct single error and detect double errors. When double errors are detected, HARQ scheme is used to correct the errors. These works have focused only to correct one bit (or) two bit random errors. But, NoC interconnect wires are more vulnerable to multiple random and burst errors because of DSM noise. Hence, more power full error correction schemes are needed to correct multiple random errors as well as burst errors. In adaptive error control schemes have been proposed for variable noise environment. In this work, single SEC Hamming code is used to correct single random error in low noise environment and multiple SEC Hamming code to correct multiple random errors in high noise environment. The author has also used multiple SEC Hamming code with interleaving to correct burst errors in high noise environment. The author use single SEC Hamming code to correct single random error in low noise environment and Hamming product codes with type II HARQ to correct multiple random errors and burst errors in high noise environment. This code does not include crosstalk avoidance with it. Joint crosstalk avoidance and single error/multiple random error correction codes have been proposed. Crosstalk avoidance and single error correction (CAC/SEC) like duplicate add parity (DAP), dual rail (DR), boundary shift code (BSC), modified dual rail (MDR)and triplication error correction scheme[29], reduce the coupling capacitance of the on chip interconnect wire from (1 + 4k)CLto (1 + 2k)CL and simultaneously corrects single random error. The authors propose crosstalk avoidance SEC and two bit burst error detection. When two bit burst error is detected, HARQ retransmission scheme is used to correct the errors. DAP coding scheme is used for triple error correction and quadruple error detection. Triplication error correction coding scheme and majority decoder are used to correct only single random error. Previous works proposed for combined crosstalk avoidance and error correction code, have focused only to correct errors of maximum three bits only. The works use DAP coding scheme and correct only one, two or three bits errors with crosstalk avoidance. But, the work proposed in this paper corrects any error pattern up to five (i.e. 1, 2, 3, 4, and 5 errors) including combination of random and burst errors and simultaneously avoiding crosstalk between interconnect wires.

III. PROPOSED SYSTEM

As discussed in the introduction, the goal is to design SEC codes that can protect a data block plus a few control bits such that the control bits can be decoded with low delay. As mentioned before, the data blocks to be protected have a size that is commonly a power of two, e.g., 64 or 128 bits. To protect a 64-bit data block with a SEC code, 7 parity check bits are needed, while 8 are enough to protect 128 bits. In the first case, there are2^7=128possible syndromes, and therefore, the SEC code can be extended to cover a few additional control bits. The same is true for 128 bits and, in general, for a SEC code that protects a data block that is a power of two. This means that the control bits can also be protected with no additional parity check bits. This is more efficient than using two separate SEC codes (one for the data bits and the other for the control bits) as this requires additional parity check bits. The main problem in using an extended SEC code is that the decoding of the control bits is more complex. To illustrate this issue, let us consider a 128-bit data block and 3 control bits. The initial SEC code for the 128-bit data block has the parity check matrix The decoding of a bit in each case is shown in Fig. 4, and the difference in complexity is apparent. As discussed earlier, our goal is to simplify the decoding of the control bits while using a single SEC code for both data and control bits. To do so, the first step is to note that, in some cases, SEC decoding can be simplified to check only some of the syndrome bits. One example is the decoding of constant-weight SEC codes proposed. In this case, only the syndrome bits that have a 1 in the column of the parity check matrix need to be checked.

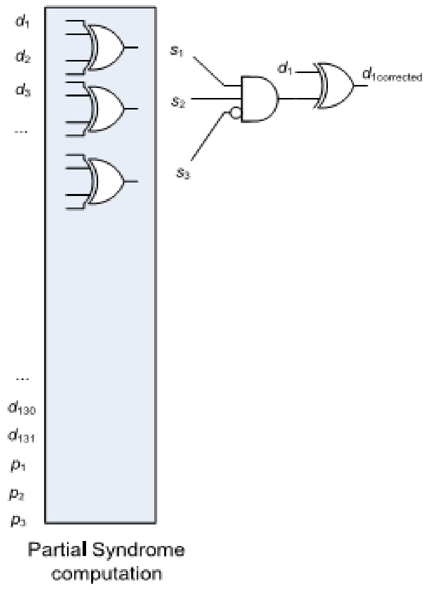


Fig6. Bit decoding of a control bit in the proposed SEC code

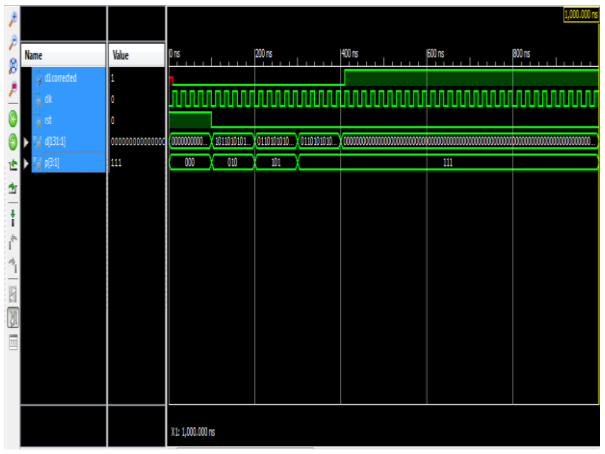
This simplifies the decoding for all bits but, in most cases, requires additional parity check bits. In our case, the main focus is to simplify the decoding of the control bits as those are commonly on the critical path. To do so, the parity check bits can be divided in two groups: a first group that is shared by both data and control bits and a second that is used only for the data bits. Then, the decoding of the control bits only requires the re-computation of the first group of parity check bits. This scheme is better illustrated with an example. Let us consider a 128-bit data block and 3 control bits protected with 8 parity check bits. Those 8 bits are divided in a group of 3 shared between data and control bits and a second group of 5 that is used only for the data bits. To protect the control bits, the first three parity check bits can be assigned different values for each control bit, and the remaining parity check bits are not used to protect the control bits.

The rest of the values are used to protect the data bits, and for each value, different values of the remaining five parity check bits can be used. In this example, the first group has 3 bits that can take 8 values, and three of them are used for the columns that correspond to the control bits. This leaves 5 values that can be used to protect the data bits. The second group of parity check bits has 5 bits that can be used to code 32 values for each of the 5 values on the first group. Therefore, a maximum of5×32 = 160data bits can be protected. In fact, the number is lower as the zero value on the first group cannot be combined with a zero or a single one on the second group as the corresponding column would have weight of zero or one. In any case, 128 data bits can be easily protected. An example of the parity check matrix of a SEC code derived using this method. The three first columns correspond to the added control bits. The two groups of parity check bits are also separated, and the first three rows are shared for data and control bits, while the last five only protect the data bits. It can be observed that the control bits can be decoded by simply re-computing the first three parity check bits. In addition, the zero value on these three bits is also used for some data bits. This means that those bits are not needed to re-compute the first three parity check bits. The decoding of one of the control bits is illustrated in Fig. 6.It can be observed that the circuitry is significantly simpler than that of a traditional SEC code.

This will be confirmed by the\ experimental results presented in the next section. The method can also be used to protect more than three control bits. In a general case, let us consider that we need to protect d data bits and c control bits using p parity check bits. Then, p is divided in two groups pcd and pd. The first group is shared between control and data bits, and the second is used only for the data bits. The number of data bits that can be protected with this scheme can be calculated as follows. The number of combinations of the first group available to be used to protect the data bits is2 Pcd−c. For each of those, up to 2 Pd values can be used, giving total of(2 Pcd−c)·2 Pd .However, for the zero value, the combinations of the second group with weight zero or one cannot be used, sopd+1 should be subtracted. Similarly, for the pcd values with weight one on the first group, the zero value on the second group cannot be used as the resulting column would have weight one. Therefore, pcd should also be subtracted, giving a total of (2 Pcd−c)·2 Pd−(pd+1)−pcd. This is the number of data bits that can be protected in addition to the control bits. As the number of control bits increases, pcd must also be increased to be able to protect the block of data bits with the same number of parity check bits. Increasing pcd makes the decoding of control bits more complex; therefore, the minimum value should be used

IV. RESULTS

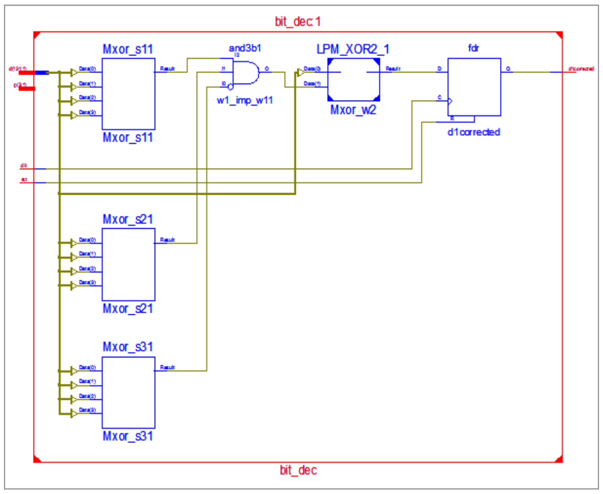
SIMULATION RESULT



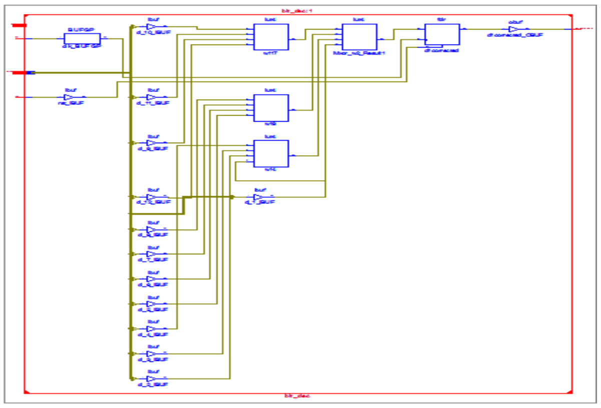
SYNTHESIS RESULTS

The developed project is simulated and verified their functionality. Once the functional verification is done, the RTL model is taken to the synthesis process using the Xilinx ISE tool. In synthesis process, the RTL model will be converted to the gate level netlist mapped to a specific technology library.

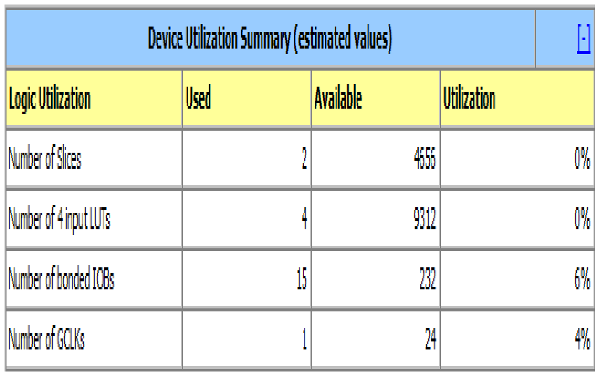
RTL SCHEMATIC



TECHNOLOGY SCHEMATIC



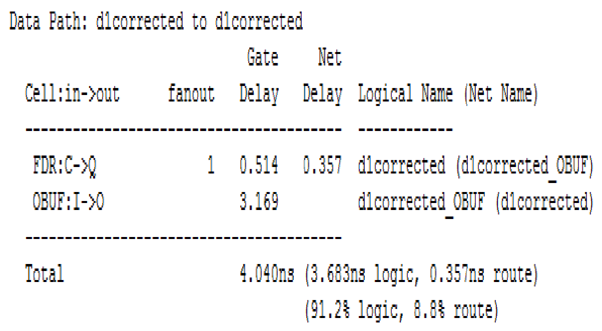
DESIGN SUMMARY



Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	2	4656	0%
Number of 4 input LUTs	4	9312	0%
Number of bonded IOBs	15	232	6%
Number of GCLKs	1	24	4%

TIMING REPORT

e-ISSN: 2348-6848
p-ISSN: 2348-795X
Volume 05 Issue 07
March 2018



```
Data Path: d1corrected to d1corrected
                                 Gate     Net
  Cell:in->out      fanout   Delay   Delay  Logical Name (Net Name)
  ----------------------------------------  ------------
   FDR:C->Q              1   0.514   0.357  d1corrected (d1corrected_OBUF)
   OBUF:I->O                 3.169          d1corrected_OBUF (d1corrected)
  ----------------------------------------
  Total                      4.040ns (3.683ns logic, 0.357ns route)
                                     (91.2% logic, 8.8% route)
```

V CONCLUSION

In this brief, a method to construct SEC codes that can protect a block of data and some additional control bits has been presented. The derived codes are designed to enable fast decoding of the control bits. The derived codes have the same number of parity check bits as existing SEC codes and therefore do not require additional cost in terms of memory or registers. To evaluate the benefits of the proposed scheme, several codes have been implemented and compared with minimum-weight SEC codes. The proposed codes are useful in applications, where a few control bits are added to each data block and the control bits have to be decoded with low delay. This is the case on some networking circuits. The scheme can also be useful in other applications where the critical delay affects some specific bits such as in some finite-state machines. Another example is arithmetic circuits where the critical path is commonly on the least significant bits. Therefore, reducing the delay on those bits can increase the overall circuit speed.

REFERENCES

[1] P. Bosshart et al., “Forwarding metamorphosis: Fast programmable match-action processing in hardware for SDN,” inProc. SIGCOMM,
2013, pp. 99–110.

[2] J. W. Lockwood et al., “NetFPGA—An open platform for gigabit-ratenetwork switching and routing,” inProc. IEEE Int. Conf. Microelectron.Syst. Educ., Jun. 2007, pp. 160–161.

[3] A. L. Silburt, A. Evans, I. Perryman, S.-J. Wen, and D. Alexandrescu,“Design for soft error resiliency in Internet core routers,”IEEE Trans.Nucl. Sci., vol. 56, no. 6, pp. 3551–3555, Dec. 2009.

[4] E. Fujiwara,Code Design for Dependable Systems: Theory and PracticalApplication. Hoboken, NJ, USA: Wiley, 2006.

[5] C. L. Chen and M. Y. Hsiao, “Error-correcting codes for semiconductormemory applications: A state-of-the-art review,”IBM J. Res. Develop.,vol. 28, no. 2, pp. 124–134, Mar. 1984.

[6] V. Gherman, S. Evain, N. Seymour, and Y. Bonhomme, “Generalizedparity-check matrices for SEC-DED codes with fixed parity,” inProc.IEEE On-Line Test. Symp., 2011, pp. 198–20.