# Integer Codes Correcting Sparse Byte Errors

Aleksandar Radonjic* and Vladimir Vujicic

Institute of Technical Sciences of the Serbian Academy of Sciences and Arts, Belgrade, Serbia
E-mails: sasa_radonjic@yahoo.com, vujicicv@yahoo.com (*Corresponding author)

***Abstract*: In public optical networks, the data are scrambled with a $x^u + 1$ self-synchronous scramblers (SSSs). The reason for this is to avoid long strings of ones or zeros, which might affect the receiver synchronization. Unfortunately, the use of SSSs is always related to the problem of duplication of channel errors. More precisely, each error occurring during the transmission will be duplicated *u* bits later. In this paper, we present a low-cost solution to this problem based on integer codes capable of correcting sparse byte errors.**

***Keywords*: Integer codes, sparse byte errors, error correction, public optical networks.**

## 1. Introduction

In many communication networks the data are randomized before transmission. The reason for this is to avoid long strings of 1s or 0s, which might affect the receiver synchronization. In public optical networks (PONs), such as synchronous optical network (SONET) and high-level data link control (HDLC) [1], the process of randomization is performed using self-synchronous scramblers (SSSs). These devices modify the data by XORing two bits that are spaced *u* bits apart ($u$ = 29 or 43) [6]-[10]. Such sequence is then sent to the receiver, which performs the same operation to recover the original unscrambled data.

Although this procedure is simple to implement, it has a drawback of error duplication. In other words, each error occurring during the transmission will be duplicated *u* bits later. In mentioned networks, where random errors dominate [2]-[5], this will cause the appearance of two errors which cannot be corrected by standard cyclic redundancy check (CRC) codes. In order to overcome this problem, the researchers proposed the use of modified CRC codes (m-CRCCs) [6]-[10]. These codes were preferred over other codes (e.g. BCH codes) due to their lower decoding complexity. In practice, however, such a solution would be complex to implement, since it is also based on the modification of existing network hardware (SONET terminals, HDLC controllers, etc.).

Bearing this in mind, in this paper, we present a class of integer codes that are suitable for use in modern PONs. Compared to m-CRCCs, these codes have three advantages. First, they use integer and lookup table operations, which are supported by all processors [11]. As a result, they can be implemented "for free" (in software), i.e. without modifying the network hardware. The second advantage is that the proposed codes can correct three types of errors within a *b*-bit byte:

single errors, double errors and triple-adjacent errors. Hence, they are more powerful than the codes suggested in [6]-[10]. Finally, the proposed codes can be interleaved without delay and without using dedicated hardware. Thanks to this, it is possible to construct simple codes capable of correcting errors affecting several consecutive bytes.

The organization of this paper is as follows: Section 2 deals with the construction of integer codes capable of correcting sparse byte (SB) errors. The error control procedure and theoretical decoding throughputs for these codes are described and evaluated in Section 3. Finally, Section 4 concludes the paper.

## 2. Codes Construction

In this section, we start with four definitions that are related to the construction of integer codes capable of correcting SB errors.

**Definition 1.** *An error is called a SB error if, within a b-bit byte, one, two random or three adjacent bits are in error.*

**Definition 2.** *Let* $Z_{2^b-1} = \{0, 1, \ldots, 2^b - 2\}$ *be the ring of integers modulo* $2^b - 1$ *and let* $B_i = \sum_{n=0}^{b-1} a_n \cdot 2^n$ *be the integer representation of a b-bit byte, where* $a_n \in \{0, 1\}$ *and* $1 \le i \le k$. *Then, the code C(b, k, c), defined as*

$$C(b,k,c) = \left\{ (B_1, B_2, \ldots, B_k, B_{k+1}) \in Z_{2^b-1}^{k+1} \colon \sum_{i=1}^{k} C_i \cdot B_i \equiv B_{k+1} \ (\mathrm{mod}\ 2^b - 1) \right\} \tag{1}$$

*is an* $(kb + b, kb)$ *integer code, where* $c = (C_1, C_2, \ldots, C_k, 1) \in Z_{2^b-1}^{k+1}$ *is the coefficient vector and* $B_{k+1} \in Z_{2^b-1}$ *is an integer.*

**Definition 3.** *Let* $x = (B_1, B_2, \ldots, B_k, B_{k+1}) \in Z_{2^b-1}^{k+1}$, $y = (\underline{B}_1, \underline{B}_2, \ldots, \underline{B}_k, \underline{B}_{k+1}) \in Z_{2^b-1}^{k+1}$ *and* $e = y - x = (\underline{B}_1 - B_1, \underline{B}_2 - B_2, \ldots, \underline{B}_k - B_k, \underline{B}_{k+1} - B_{k+1}) = (e_1, e_2, \ldots, e_k, e_{k+1}) \in Z_{2^b-1}^{k+1}$ *be respectively, the sent codeword, the received codeword and the error vector. Then, the syndrome S of the received codeword is defined as*

$$S = \sum_{i=1}^{k} (C_i \cdot \underline{B}_i - \underline{B}_{k+1}) \ (\mathrm{mod}\ 2^b - 1) = \sum_{i=1}^{k+1} e_i \cdot C_i \ (\mathrm{mod}\ 2^b - 1) \tag{2}$$

**Definition 4.** *An* $(kb + b, kb)$ *integer code is called SB error correcting (SBEC) if it can correct error vectors from the set* $E = \{(e_i, 0, \ldots, 0, 0), \ldots, (0, 0, \ldots, e_i, 0), (0, 0, \ldots, 0, e_i)\}$, *where* $e_i \in \{\pm 2^r, \pm 2^s \pm 2^t, (\pm 2^2 \pm 2^1 \pm 2^0)\cdot 2^m\}$, $0 \le r \le b - 1$, $0 \le s < t \le b - 1$ *and* $0 \le m \le b - 3$.

**Definition 5.** *The error set for an* $(kb + b, kb)$ *integer SBEC code is defined by*

$$\xi = s_1 \cup s_2 \cup s_3 \tag{3}$$

*where*

$$s_1 = \left\{ \bigcup_{i=1}^{k+1} \left( \pm 2^r \cdot C_i \right) \left( \mathrm{mod}\ 2^b - 1 \right) \colon 0 \le r \le b - 1 \right\}, \tag{4}$$

$$s_2 = \left\{ \bigcup_{i=1}^{k+1} \left[ \left( \pm 2^r \pm 2^s \right) \cdot C_i \right] \left( \bmod 2^b - 1 \right) : 0 \le r < s \le b-1 \right\}, \tag{5}$$

$$s_3 = \left\{ \bigcup_{i=1}^{k+1} \left[ \left( \pm 2^2 \pm 2^1 \pm 2^0 \right) \cdot 2^m \cdot C_i \right] \left( \bmod 2^b - 1 \right) : 0 \le m \le b-3 \right\}. \tag{6}$$

With these definitions, we are ready to state the following theorem.

**Theorem 1**. *The sets $s_1$ and $s_3$ are subsets of $s_2$.*

**Proof.** An element $\alpha$ of $s_1$ takes the form $(\pm 2^r \cdot C_i) \pmod{2^b - 1}$, where $0 \le r \le b - 1$. The set $s_2$ contains elements $\beta$ taking the form $[(\pm 2^r \pm 2^s) \cdot C_i] \pmod{2^b - 1}$, where $0 \le r < s \le b - 1$. Obviously, if $s = r + 1$, $\beta$ will take two forms: $\beta_1 = (\pm 2^r \cdot C_i) \pmod{2^b - 1}$ and $\beta_2 = (\pm 3 \cdot 2^r \cdot C_i) \pmod{2^b - 1}$, where $0 \le r \le b - 2$. On the other hand, if $r = 0$ and $s = b - 1$, $\beta$ will take the form $\beta_3 = [(\pm 1 \pm 2^{b-1}) \cdot C_i] \pmod{2^b - 1} = (\mp 2^{b-1} \cdot C_i) \pmod{2^b - 1}$. For this it is easy to conclude that $\alpha = \beta_1 \cup \beta_3$, i.e. that $s_1 \subseteq s_2$. Similarly, an element $\gamma$ of $s_3$ takes the form $[(\pm 2^2 \pm 2^1 \pm 2^0) \cdot 2^m \cdot C_i] \pmod{2^b - 1}$, where $0 \le m \le b - 3$. Note that this element is of the form $(\pm \delta \cdot 2^m \cdot C_i) \pmod{2^b - 1}$, where $\delta \in \{1, 3, 5, 7\}$. Now, if $\delta = 1$, then $\gamma_1 = (\pm 2^m \cdot C_i) \pmod{2^b - 1} \in \alpha \subseteq \beta$. If $\delta = 3$, then $\gamma_2 = (\pm 3 \cdot 2^m \cdot C_i) \pmod{2^b - 1} = [\pm (2^{m+1} + 2^m) \cdot C_i] \pmod{2^b - 1} \in \beta$. If $\delta = 5$, then $\gamma_3 = (\pm 5 \cdot 2^m \cdot C_i) \pmod{2^b - 1} = [\pm (2^{m+2} + 2^m) \cdot C_i] \pmod{2^b - 1} \in \beta$. Finally, if $\delta = 7$, then $\gamma_4 = (\pm 7 \cdot 2^m \cdot C_i) \pmod{2^b - 1} = [\pm (2^{m+3} - 2^m) \cdot C_i] \pmod{2^b - 1} \in \beta$. In all cases, $\gamma \subseteq \beta$. Hence, $s_3 \subseteq s_2$. □

Now we can prove the main theorem of this section.

**Theorem 2**. *The codes defined by (1) can correct all SB errors only if there exist k mutually different coefficients $C_i \in Z_{2^b-1} \setminus \{0,1\}$ such that*

$$|\xi| = |s_2| = \left[ 2 \cdot (b-1)^2 - 2 \right] \cdot (k+1),$$

*where $|A|$ is the cardinality of $A$.*

**Proof.** To prove the theorem, observe that the set $s_2$ can be express as the union

$$s_2 = \bigcup_{j=1}^{2b-2} R_j$$

where

$$R_1 = \left\{ \left[ \pm (2^1 - 2^0) \cdot 2^r \cdot C_i \right] (\bmod 2^b - 1) : 0 \le r \le b-2,\ 1 \le i \le k+1 \right\}$$

$$R_2 = \left\{ \left[ \pm (2^1 + 2^0) \cdot 2^r \cdot C_i \right] (\bmod 2^b - 1) : 0 \le r \le b-2,\ 1 \le i \le k+1 \right\}$$

$$R_3 = \left\{ \left[ \pm (2^2 - 2^0) \cdot 2^r \cdot C_i \right] (\bmod 2^b - 1) : 0 \le r \le b-3,\ 1 \le i \le k+1 \right\}$$

$$R_4 = \left\{ \left[ \pm (2^2 + 2^0) \cdot 2^r \cdot C_i \right] (\bmod 2^b - 1) : 0 \le r \le b-3,\ 1 \le i \le k+1 \right\}$$

$$R_5 = \left\{ \left[ \pm (2^3 - 2^0) \cdot 2^r \cdot C_i \right] (\bmod 2^b - 1) : 0 \le r \le b-4,\ 1 \le i \le k+1 \right\}$$

$$R_6 = \left\{ \left[ \pm (2^3 + 2^0) \cdot 2^r \cdot C_i \right] (\bmod 2^b - 1) : 0 \le r \le b-4,\ 1 \le i \le k+1 \right\}$$

$$\vdots$$

$$R_{2b-7}=\left\{\left[\pm(2^{b-3}-2^0)\cdot 2^r\cdot C_i\right] \pmod{2^b-1} : 0\le r\le 2,\ 1\le i\le k+1\right\}$$
$$R_{2b-6}=\left\{\left[\pm(2^{b-3}+2^0)\cdot 2^r\cdot C_i\right] \pmod{2^b-1} : 0\le r\le 2,\ 1\le i\le k+1\right\}$$
$$R_{2b-5}=\left\{\left[\pm(2^{b-2}-2^0)\cdot 2^r\cdot C_i\right] \pmod{2^b-1} : 0\le r\le 1,\ 1\le i\le k+1\right\}$$
$$R_{2b-4}=\left\{\left[\pm(2^{b-2}+2^0)\cdot 2^r\cdot C_i\right] \pmod{2^b-1} : 0\le r\le 1,\ 1\le i\le k+1\right\}$$
$$R_{2b-3}=\left\{\left[\pm(2^{b-1}-2^0)\cdot 2^r\cdot C_i\right] \pmod{2^b-1} : r=0,\ 1\le i\le k+1\right\}$$
$$R_{2b-2}=\left\{\left[\pm(2^{b-1}+2^0)\cdot 2^r\cdot C_i\right] \pmod{2^b-1} : r=0,\ 1\le i\le k+1\right\}$$

The syndromes caused by SB errors will be nonzero and mutually different only if there exists $k$ different coefficients $C_i\in Z_{2^b-1}\setminus\{0,1\}$ such that

$$R_1\cap R_2\cap\cdots\cap R_{2b-2}=\varnothing,$$
$$\left|R_{2t-1}\right|=\left|R_{2t}\right|=2\cdot(b-t)\cdot(k+1),\ t=1,2,...,b-1.$$

Further, if we compare the sets $R_2$, $R_3$, $R_{2b-2}$ and $R_{2b-5}$ we can note that $R_{2b-5}\subseteq\left(R_{2b-2}\cup R_2\right)$ and $R_3\subseteq R_2$. As a result, it follows that

$$\left|s_2\right|=\sum_{j=1}^{2b-2}\left|R_j\right|-\left|R_3\right|-\left|R_{2b-5}\right|=\left[4\cdot\sum_{j=1}^{b-1}(b-j)-(b-2)-2\right]\cdot(k+1)=\left[2\cdot(b-1)^2-2\right]\cdot(k+1).$$

Conversely, if the codes satisfy the above condition, then we correct all SB errors. Therefore, these codes are ($kb$ + $b$, $kb$) integer SBEC codes. □

Now, by knowing the cardinality of $s_2$, we can derive the upper bound on code length.

**Theorem 3.** *For any* ($kb$ + $b$, $kb$) *integer SBEC code it holds that*

$$k\le\left\lfloor\frac{2^{b-1}-(b-1)^2}{(b-1)^2-1}\right\rfloor.$$

**Proof.** From Definition 1 we know that the total number of nonzero syndromes is $2^b-2$. In addition, from Theorem 2 we know that the set $s_2$ has $\left[2\cdot(b-1)^2-2\right]\cdot(k+1)$ nonzero elements. Consequently, we have the inequality

$$\left[2\cdot(b-1)^2-2\right]\cdot(k+1)\le 2^b-2$$

wherefrom it follows that

$$k\le\left\lfloor\frac{2^{b-1}-(b-1)^2}{(b-1)^2-1}\right\rfloor.\ \square$$

To illustrate the applicability of Theorems 2-3 we have conducted an exhaustive computer search. Our first goal was to compare the obtained results with the theoretical bounds (Table 1), while the second goal was finding the coefficients $C_i$ (Table 2) for 32-bit codes (these codes are perfectly suited for implementation on modern 32/64-bit processors [11]).

**Table 1**. Number of Coefficients for Some Integer SBEC Codes.

| | $b = 8$ | $b = 9$ | $b = 10$ | $b = 11$ | $b = 12$ | $b = 13$ | $b = 14$ | $b = 15$ | $b = 16$ |
|---|---|---|---|---|---|---|---|---|---|
| Theoretical bound | 1 | 3 | 5 | 9 | 16 | 27 | 47 | 83 | 145 |
| Computer-search result | 0 | 1 | 1 | 3 | 6 | 10 | 16 | 27 | 43 |

**Table 2**. First 128 Coefficients in [2, $2^{32}$ - 2] for 32-bit Integer SBEC Codes.

| | | | | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 19 | 23 | 25 | 27 | 29 | 37 | 39 | 41 | 47 | 49 | 53 | 59 | 61 | 67 | 71 | 77 |
| 79 | 83 | 89 | 97 | 101 | 103 | 107 | 109 | 113 | 121 | 131 | 137 | 139 | 149 | 151 | 157 |
| 163 | 167 | 173 | 179 | 181 | 191 | 193 | 197 | 199 | 211 | 223 | 227 | 229 | 233 | 239 | 251 |
| 263 | 269 | 271 | 277 | 281 | 283 | 289 | 293 | 307 | 311 | 313 | 317 | 331 | 337 | 347 | 349 |
| 353 | 357 | 359 | 361 | 365 | 367 | 373 | 379 | 383 | 389 | 397 | 401 | 409 | 419 | 421 | 431 |
| 433 | 437 | 439 | 443 | 449 | 457 | 461 | 463 | 465 | 467 | 475 | 479 | 487 | 491 | 499 | 503 |
| 521 | 523 | 529 | 541 | 547 | 551 | 557 | 563 | 569 | 571 | 575 | 577 | 587 | 593 | 599 | 601 |
| 607 | 613 | 617 | 619 | 621 | 625 | 631 | 641 | 643 | 647 | 653 | 659 | 661 | 667 | 673 | 675 |

## 3. Error Control Procedure and Theoretical Decoding Throughputs

The error control procedure for the proposed codes is similar to that described in [12]-[18]. In short, it consists of two steps: obtaining the error correction data from the syndrome table and executing the operation

$$B_i = \underline{B}_i - e \pmod{2^b - 1} \qquad (7)$$

where $1 \le i \le k + 1$, $e = \pm 2^r \pm 2^s$ and $0 \le r < s \le b - 1$. To generate the syndome table it is necessary to substitute the values of $b$ and $C_i$ (Table 2) into (5). In this way, exactly $|\xi|$ (Theorem 2) relationships between the syndrome (element of the set $\xi$), error location ($i$) and error vector ($e$) are established (Fig. 1). Accordingly, when $S \neq 0$, the decoder's task is to find the entry with the first $b$ bits as that of the syndrome $S$. If the elements of $\xi$ are sorted in increasing order, this task will be completed after $n_{\mathrm{TL}}$ table lookups, where $1 \le n_{\mathrm{TL}} \le \lfloor log_2 |\xi| \rfloor + 2$ [19].

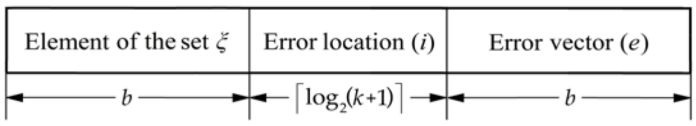

**Fig. 1**. Bit-width of one syndrome table entry.

To illustrate the effectiveness of the above approach, suppose that the data packet has $K = 6 \cdot b \cdot k = 192 \cdot k$ data bits ($k$ = 32, 64, 96 and 128) and that each network node is equipped with the six-core processor (Fig. 2) having the following parameters [20]:

1) clock rate: $C_{\mathrm{R}} = 3.3 \cdot 10^9$ Hz,

2) integer addition/subtraction latency: 1 cycle,

3) integer multiplication latency: 3 cycles,

4) modulo reduction operation: 1 cycle latency,

5) 32-bit comparison operation: 1 cycle latency,

6) 128-bit shift operation latency: 1 cycle,

7) L1 cache (32 KB per core) access latency: 4 cycles,

8) L2 cache (256 KB per core) access latency: 12 cycles,

9) L3 cache (15 MB shared) access latency: 34 cycles.

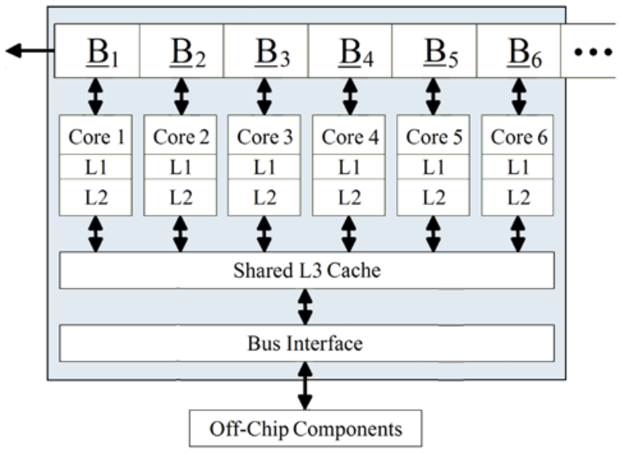

**Fig. 2.** Block diagram of a six-core processor.

In addition, let us assume that the coefficients $C_i$ (Table 2) are stored in each of the six L1 caches and that the syndrome table is placed into the L3 cache (Fig. 2). In that case, instead of one, the decoder (processor) will (in parallel) compute the values of six syndromes:

- *Core* 1

$$S_1 = \sum_{i=1}^{k} (C_i \cdot \underline{B}_{6\cdot(i-1)+1} - \underline{B}_{6\cdot k+1}) \ (\mathrm{mod}\ 2^{32} - 1) \tag{8}$$

- *Core* 2

$$S_2 = \sum_{i=1}^{k} (C_i \cdot \underline{B}_{6\cdot(i-1)+2} - \underline{B}_{6\cdot k+2}) \ (\mathrm{mod}\ 2^{32} - 1) \tag{9}$$

$\vdots$

- *Core* 6

$$S_6 = \sum_{i=1}^{k} (C_i \cdot \underline{B}_{6\cdot(i-1)+6} - \underline{B}_{6\cdot k+6}) \ (\mathrm{mod}\ 2^{32} - 1) \tag{10}$$

If we add to this $K/128 = 1.5\cdot k$ shift operations, we see that each core requires $\mathrm{T}_1 = 9.5\cdot k + 1$ clock cycles ($k$ accesses to the L1 cache, $k$ integer multiplications, $k$ - 1 integer additions, $1.5\cdot k$ shift operations, 1 integer subtraction and 1 modulo reduction) to calculate the values of all syndromes. If one or more syndromes are non-zero, the decoder will additionally perform $n_{\mathrm{TL}}$ table lookups, $n_{\mathrm{TL}}$ comparisons, 1 integer addition and 1 modulo reduction. In our case, six such operations can be executed in parallel in $\mathrm{T}_2 = 35\cdot n_{\mathrm{TL}} + 2$ clock cycles. So, if we sum up both processing times, we come to the conclusion that the decoder requires

$$\mathrm{T}_{\mathrm{total}} = \mathrm{T}_1 + \mathrm{T}_2 = 9.5\cdot k + 35\cdot n_{\mathrm{TL}} + 3 \tag{11}$$

clock cycles to process $K$ data bits, i.e. one second to decode

$$G = \frac{C_R}{T_{total}/K} = \frac{(3.3\cdot10^9)\cdot K}{9.5\cdot k + 35\cdot n_{TL} + 3} = \frac{(3.3\cdot10^9)\cdot192\cdot k}{9.5\cdot k + 35\cdot n_{TL} + 3} \quad (12)$$

data bits. From (12) it is easy to calculate that the theoretical throughput of the decoder varies between 22.48 Gbps and 43.05 Gbps (Table 3). This means that all codes from Table 3 have the potential to be used in 10G networks (e.g. 10G SONET and HDLC network) [1]. Besides this, from Table 3 we see that the code with the code rate 4096/4128 has theoretical throughput above 40 Gbps. This fact makes it a good candidate for use in 40G networks (e.g. 40G SONET) [1]. Finally, from (8)-(10) we see that the analyzed codes are interleaved at the byte level. Thanks to this, they are able both to protect up to 24576 bits and to correct SB errors spanning up to 192 bits. Such solution is not only more reliable than [6]-[10], but also much simpler to implement (Table 4).

**Table 3**. Memory requirements and theoretical decoding throughputs for some six-byte interleaved integer SBEC codes.

| Code | $k$ | Memory requirements for storing the coefficients $C_i$ | Memory requirements for storing the syndrome table | Number of table lookups | Minimum theoretical decoding throughput |
|---|---|---|---|---|---|
| (1056, 1024) | 32 | 6 x 128 B | 0.55 MB | $1 \le n_{TL} \le 17$ | 22.48 Gbps |
| (2080, 2048) | 64 | 6 x 256 B | 1.11 MB | $1 \le n_{TL} \le 18$ | 32.68 Gbps |
| (3104, 3072) | 96 | 6 x 384 B | 1.65 MB | $1 \le n_{TL} \le 19$ | 38.50 Gbps |
| (4128, 4096) | 128 | 6 x 512 B | 2.23 MB | $1 \le n_{TL} \le 19$ | 43.05 Gpbs |

**Table 4**. Comparison of codes used or proposed for use in modern PONs.

| Main characteristics | CRC codes | Modified CRC Codes [6]-[10] | Proposed codes (interleaved version) |
|---|---|---|---|
| Error control capabilities | Detection of single, double, triple and burst errors of lenght up to 16 bits | Correction of single and duplicate errors, and detection of double and triple errors | Correction of SB errors affecting several consecutive *b*-bit bytes |
| Equal error protection | No (packet header only) | No (packet header only) | Yes |
| Processing of data bits | Modulo-2 operations | Modulo-2 operations | Integer and lookup table operations |
| Type of implementation | Hardware | Hardware | Software |
| Universal application | Yes (any PON) | No (specific PON) | Yes (any PON) |

## 4. Conclusion

In this paper, we presented a new class of integer error control codes. We have shown that these codes have three characteristics: first, they can correct sparse byte errors, second, they operate under integer arithmetic, and third, they can be interleaved without delay and without using additional hardware. Thanks to these features, the presented codes are well suited to be used in practice, especially in optical networks such as SONET and HDLC.

## References

[1] R. Ramaswani, K. Sivarajan and G. Sasaki, *Optical Networks: A Practical Perspective,* 3rd ed., Elsevier, Inc., 2010.

[2] CCITT Study Group XVIII Contribution D21, "Observations of Error Characteristics of Fiber Optic Transmission Systems," Jan. 1989.

[3] W. Grover and D. Moore, "Design and Characterization of an Error-Correcting Code for the SONET STS-1 Tributary," *IEEE Trans. Commun.*, vol. 38, pp. 467-476, Apr. 1990.

[4] W. Grover, "Effect of Error Correcting Code Using D-S3 Framing Bits on Measured Dribble Error Pattern of 565 Mb/s Fibre Optic Transmission System", *Elect. Lett.*, vol. 28, no. 20, pp. 1869-1870, Sept. 1992.

[5] M. Cheung, W. Grover and W. Krzymien, "Combined Framing and Error Correction Coding for DS3 Signal Format," *IEEE Trans. Commun.*, vol. 43, nos. 2-4, pp. 1365-1374, Feb-Apr. 1995.

[6] N. Figueira, "Networking Device and Method for Making Cyclic Redundancy Check (CRC) Immune to Scrambler Error Duplication," U.S. Patent 6,609,226, Aug. 19, 2003.

[7] S. Gorshe, "Cyclic Redundancy Check Circuit for Use with Self-Synchronous Scramblers," U.S. Patent 7,353,446, Nov. 17, 2005.

[8] S. Gorshe, "Analysis of the Interaction Between Linear Cyclic Error Correcting Codes and Self-Synchronous Payload Scramblers", *IEEE Trans. Commun.*, vol. 56, no. 11, pp. 1800-1806, Nov. 2008.

[9] S. Gorshe, "Forward Error Correction with Self-Synchronous Scramblers," U.S. Patent 7,913,151, Mar. 22, 2011.

[10] D. Ferguson *et al.*, "System, Apparatus, and Method for Increasing Resiliency in Communications," U.S. Patent 7,986,717, Jul. 26, 2011.

[11] R. Giladi, *Network Processors: Architecture, Programming, and Implementation*, Elsevier, Inc., 2008.

[12] A. Radonjic and V. Vujicic, "Integer Codes Correcting Burst Errors within a Byte," *IEEE Trans. Comput.*, vol. 62, no. 2, pp. 411-415, Feb. 2013.

[13] A. Radonjic *et al.*,"Integer Codes Correcting Double Asymmetric Errors," *IET Commun.*, vol. 10, no. 14, pp. 1691-1696, Sep. 2016.

[14] A. Radonjic and V. Vujicic, "Integer Codes Correcting Spotty Byte Asymmetric Errors," *IEEE Commun. Lett.*, vol. 20, no. 12, pp. 2338-2341, Dec. 2016.

[15] A. Radonjic and V. Vujicic, "Integer Codes Correcting High-Density Byte Asymmetric Errors," *IEEE Commun. Lett.*, vol. 21, no. 4, pp. 694-697, Apr. 2017.

[16] A. Radonjic and V. Vujicic, "Integer Codes Correcting Single Errors and Burst Asymmetric Errors within a Byte," *Inform. Process. Lett.*, vol. 121, pp. 45-50, May 2017.

[17] A. Radonjic, "(Perfect) Integer Codes Correcting Single Errors," *IEEE Commun. Lett.*, vol. 22, no. 1, pp. 17-20, Jan. 2018.

[18] A. Radonjic and V. Vujicic, "Integer Codes Correcting Burst and Random Asymmetric Errors within a Byte," *J. Franklin Inst.*, vol. 355, no. 2, pp. 981-996, Jan. 2018.

[19] K. Mehlhorn and P. Sanders, *Algorithms and Data Structures: The Basic Toolbox*, Springer, 2008.

[20] A. Fog, "The Microarchitecture of Intel, AMD and VIA CPUs," Tech. Univ. Denmark, Denmark, Jan. 2016.