ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 1, March 2016

ERROR CORRECTION AND ERROR DETECTION CODES ON PARALLEL FILTER

U.Mathanlal, II M.E. VLSI Design, Akshaya College of Engineering and Technology, Coimbatore
mathanlaludhaya@gmail.com
Mr. A.Abdul Hayum ,Asst. Prof. (ECE Dept), Akshaya College of Engineering and Technology, Coimbatore
hayum.be08@gmail.com

Abstract-Digital filters are mostly used in signal processing and communication systems. In some cases, the reliability of those systems is extreme, and fault tolerant filter implementations are needed. Over the years, many techniques that analyze the filter structure and properties to determine fault tolerance have been proposed. As technology scales, it enables more complex systems that provide invention of many filters. In such complex systems, it is common that some of the filters function in parallel, for example, by applying the same filter to different input signals. Recently, a simple technique that uses the presence of parallel filters to achieve fault tolerance has been presented. In this brief, that idea is given brief to show that parallel filters can be protected using error correction codes (ECCs) in which each filter is the counterpart of a bit in a traditional ECC. This new scheme allows more efficient protection when the number of parallel filters is large. The technique is determined using a case study of parallel finite impulse response filters showing the effectiveness on basis of protection and implementation cost.

Index terms- ECC, filters, soft errors

I. INTRODUCTION

Electronic circuits are most probably used in automotive, medical and space applications where reliability is critical. In such applications, the circuits have to provide some degree of fault tolerance. This need is increased by the essential reliability challenges of advanced CMOS technologies that include, e.g., manufacturing variations and soft errors. More number of techniques can be used to protect a circuit from errors. Those range from modifications in the manufacturing process of the circuits to degrade the number of errors to adding redundancy at the logic or system level to ensure that errors do not make the difference to the system functionality. To add redundancy, a general technique known as triple modular redundancy (TMR) may be used. The TMR, which triplicates the design and adds voting logic to correct errors, is commonly used. However, it triples the area and power of the circuit, something that may not be acceptable in some applications. When the circuit to be prevented has algorithmic or structural properties, a better option can be to exploit those properties to determine fault tolerance. One example is signal processing circuits for which specific techniques have been proposed over the years.

Digital filters are most commonly used signal processing circuits and several techniques have been proposed to protect them from errors. Most of them have interested on finite-impulse response (FIR) filters. For example, the use of reduced precision replicas was put forward to reduce the cost of implementing modular redundancy in FIR filters. Christo Ananth et al. [5] proposed a system which contributes the complex parallelism mechanism to protect the information by using Advanced Encryption Standard (AES) Technique. AES is an encryption algorithm which uses 128 bit as a data and generates a secured data. In Encryption, when cipher key is inserted, the plain text is converted into cipher text by using complex parallelism. Similarly, in decryption, the cipher text is converted into original one by removing a cipher key. The complex parallelism technique involves the process of Substitution Byte, Shift Row, Mix Column and Add Round Key. The above four techniques are used to involve the process of shuffling the message. The complex parallelism is highly secured and the information is not broken by any other intruder.

Therefore, a significant cost reduction compared with TMR was obtained.

II.PARALLEL FILTERS WITH THE SAME RESPONSE

A discrete time filter implements the

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 1, March 2016

following equation:

$$y[n] = \sum_{l=0}^{\infty} x[n-l] \cdot h[l] \quad (1)$$

where $x[n]$ is the input signal, $y[n]$ is the output, and $h[l]$ is the impulse response of the filter [12]. When the response $h[l]$ is nonzero, only for a finite number of samples, the filter is known as a FIR filter, otherwise the filter is an infinite impulse response (IIR) filter. There are several structures to implement both FIR and IIR filters.

In the following, a set of k parallel filters with the same response and different input signals are considered. These parallel filters are illustrated in Fig. 1. This kind of filter is found in some communication systems that use several channels in parallel. In data acquisition and processing applications is also common to filter several signals with the same response.

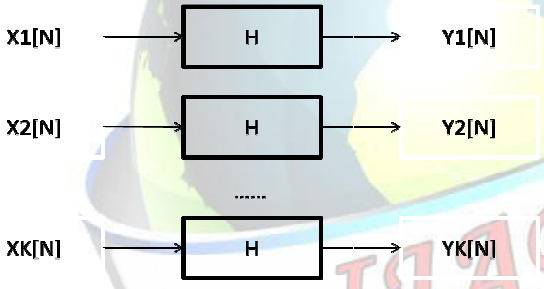

Fig 1: Parallel shifters with the same response

An interesting property for these parallel filters is that the sum of any combination of the outputs $y_i[n]$ can also be obtained by adding the corresponding inputs $x_i[n]$ and filtering the resulting signal with the same filter $h[l]$. For example

$$y_1[n] + y_2[n] = \sum_{l=0}^{\infty} x_1[n-l] + x_2[n-l] \cdot h[l]. \quad (2)$$

This simple observation will be used in the following to develop the proposed fault tolerant implementation.

III. PROPOSED SCHEME

The new technique is based on the use of the ECCs. A simple ECC takes a block of k bits and produces a block of n bits by adding $n - k$ parity check bits [13]. The parity check bits are XOR combinations of the k data bits. By properly designing those combinations it is possible to detect and correct errors. As an example, let us consider a simple Hamming code [14] with $k = 4$ and $n = 7$. The data and parity check bits are stored and can be recovered later even if there is an error in one of the bits.

s1s2s3	Error Bit Position	Action
000	No error	None
111	d1	correct d1
110	d2	correct d2
101	d3	correct d3
011	d4	correct d4
100	p1	correct p1
010	p2	correct p2
001	p3	correct p3

TABLE I: ERROR LOCATION IN THE HAMMING CODE

This is done by recomputing the parity check bits and comparing the results with the values stored. In the example considered, an error on d_1 will cause errors on the three parity checks; an error on d_2 only in p_1 and p_2; an error on d_3 in p_1 and p_3; and finally an error on d_4 in p_2 and p_3. Therefore, the data bit in error can be located and the error can be corrected. This is commonly formulated in terms of the generating G and parity check H matrixes. For the Hamming code considered in the example, those are

$$G = \begin{matrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{matrix} \quad (4)$$

$$H = \begin{matrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{matrix}. \quad (5)$$

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 1, March 2016

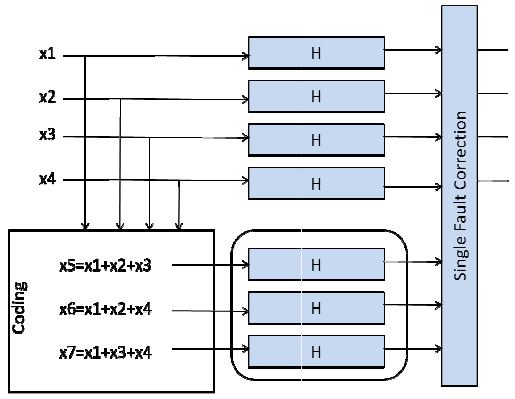

Fig 2: Proposed scheme for four filters and Hamming code

Encoding is done by computing $y = x \bullet G$ and error detection is done by computing $s = y \bullet H^T$, where the operator $\bullet$ is based on module two addition (XOR) and multiplication. Correction is done using the vector s, known as syndrome, to identify the bit in error. The correspondence of values of s to error position is captured in Table I. Once the erroneous bit is identified, it is corrected by simply inverting the bit.

$$z_1[n] = \sum_{l=0}^{\infty} x_1[n-l] + x_2[n-l] + x_3[n-l] \cdot h[l]$$

$$z_2[n] = \sum_{l=0}^{\infty} x_1[n-l] + x_2[n-l] + x_4[n-l] \cdot h[l]$$

$$z_3[n] = \sum_{l=0}^{\infty} x_1[n-l] + x_3[n-l] + x_4[n-l] \cdot h[l] \quad (6)$$

and the checking is done by testing if

$$z_1[n] = y_1[n] + y_2[n] + y_3[n]$$
$$z_2[n] = y_1[n] + y_2[n] + y_4[n]$$
$$z_3[n] = y_1[n] + y_3[n] + y_4[n]. \quad (7)$$

For example, an error on filter y_1 will cause errors on the checks of z_1, z_2, and z_3. Similarly, errors on the other filters will cause errors on a different group of z_i . Therefore, as with the traditional ECCs, the error can be located and corrected.

The overall scheme is illustrated on Fig. 2. It can be observed that correction is achieved with only three redundant filters.

For the filters, correction is achieved by reconstructing the erroneous outputs using the rest of the data and check outputs. For example, when an error on y_1 is detected, it can be corrected by making

$$y_{c1}[n] = z_1[n] - y_2[n] - y_3[n]. \quad (8)$$

and calculate $s = y\ H^T$ to detect errors. Then, the vector s is also used to identify the filter in error. In our case, a nonzero value in vector s is equivalent to 1 in the traditional Hamming code. A zero value in the check corresponds to a 0 in the traditional Hamming code.

It is important to note that due to different finite precision effects in the original and check filter implementations, the comparisons in (7) can show small differences. Those differences will depend on the quantization effects in the filter implementations that have been widely studied for different filter structures. The interested reader is referred to [12] for further details. Therefore, a threshold must be used in the comparisons so that values smaller than the threshold are classified as 0. This means that small errors may not be corrected. This will not be an issue in most cases as small errors are acceptable. The detailed study of the effect of these small errors on the signal to noise ratio at the output of the filter is left for future work. The reader can get more details on this type of analysis in [3].

	Unprotected	TMR	Method in [7]	Proposed
Slices	2914	9020	7740	6409
Flip-flops	1224	3984	3980	2941
LUTs	5692	17256	13640	12032

TABLE II: RESOURCE COMPARISON FOR FOUR PARALLEL FIR FILTERS

With this alternative formulation, it is clear that the scheme can be used for any number of parallel filters and any linear block code can be used. The approach is more attractive when the number of filters k is large. For example, when $k = 11$, only four redundant filters are needed to provide single error correction. This is the same as for traditional ECCs for which the overhead decreases as the block size increases [13].

	Unprotected	TMR	Method in [7]	Proposed

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 1, March 2016

Slices	8096	24805	21285	14422
Flip-flops	3366	10956	10945	6478
LUTs	15653	47454	37510	28331

TABLE III: RESOURCE COMPARISON FOR ELEVEN PARALLEL FIR FILTERS

The additional operations required for encoding and decoding are simple additions, subtractions, and comparisons and should have little effect on the overall complexity of the circuit. This is illustrated in Section IV in which a case study is presented.

In the discussion, so far the effect of errors affecting the encoding and decoding logic has not been considered. The encoder and decoder include several additions and subtractions and therefore the possibility of errors affecting them cannot be neglected. Focusing on the encoders, it can be seen that some of the calculations of the z_i share adders. For example, looking at (6), z_1 and z_2 share the term $y_1 + y_2$. Therefore, an error in that adder could affect both z_1 and z_2 causing a miscorrection on y_2. To ensure that single errors in the encoding logic will not affect the data outputs, one option is to avoid logic sharing by computing each of the z_i independently. In that case, errors will only affect one of the z_i outputs and according to Table I, the data outputs y_j will not be affected. Similarly, by avoiding logic sharing, single errors in the computation of the s vector will only affect one of its bits. The final correction elements such as that in (8) need to be tripled to ensure that they do not propagate errors to the outputs.

IV. CASE STUDY

To evaluate the effectiveness of the proposed scheme, a case study is used. A set of parallel FIR filters with 16 coefficients is considered. The input data and coefficients are quantized with 8 bits. The filter output is quantized with 18 bits. For the check filters z_i, since the input is the sum of several inputs x_j, the input bit-width is extended to 10 bits. A small threshold is used in the comparisons such that errors smaller than the threshold are not considered errors. As explained in Section III, no logic sharing was used in the computations in the encoder and decoder logic to avoid errors on them from propagating to the output.

Two configurations are considered. The first one is a block of four parallel filters for which a Hamming code with $k = 4$ and $n = 7$ is used. The second is a block of eleven parallel filters for which a Hamming code with $k = 11$ and $n = 15$ is used. Both configurations have been implemented in HDL and mapped to a Xilinx Virtex 4 XC4VLX80 device.

The first evaluation is to compare the resources used by the proposed scheme with those used by TMR, the protection method proposed in [7] (with $m = 7$) and by an unprotected filter implemen-tation. Those results are presented in Tables II and III for each of the configurations considered. It can be observed that the proposed technique provides significant savings (from 26% to 41%) for all the resource types (slices, flip-flops, and LUTs) compared with the TMR. The benefits are larger for the second configuration as expected with values exceeding 40% for all resource types. In that case, the relative number of added check filters $(n - k)/n$ is smaller. When compared with the arithmetic code technique proposed in [7], the savings are smaller but still significant ranging from 11% to 40%. Again, larger savings are obtained for the second configuration.

In summary, the results of this case study confirm that the proposed scheme can reduce the implementation cost significantly compared with the TMR and provides also reductions when compared with other methods such as that in [7]. As discussed before, the reductions are larger when the number of filters is large.

The second evaluation is to assess the effectiveness of the scheme to correct errors. To that end, fault injection experiments have been conducted. In particular, errors have been randomly inserted in the coefficients and inputs of the filters. In all cases, single errors were detected and corrected. In total, 8000 errors for inputs and 8000 errors for filter coefficients were inserted in the different simulation runs. This confirms the effectiveness of the scheme to correct single errors.

V. CONCLUSION

This paper has provided a new scheme to protect parallel filters that are commonly found in modern signal processing circuits. The method is based on applying ECCs to the parallel filters outputs to detect and correct errors. This technique can be given for parallel filters that have the same response

ISSN 2394-3777 (Print)
ISSN 2394-3785 (Online)
Available online at www.ijartet.com
International Journal of Advanced Research Trends in Engineering and Technology (IJARTET)
Vol. 3, Special Issue 1, March 2016

and process different input signals.

A case study has also been provided with the idea to show the effectiveness of the scheme in terms of error correction and also of circuit overheads. The technique gives larger benefits when the number of parallel filters is large. The proposed technique can also be applied to the IIR filters. In Future work the evaluation of the benefits of the proposed technique for IIR filters is considered. The extension of the scheme to parallel filters that have the same input and different impulse responses is also given for future work. The proposed scheme can also be combined with the reduced precision replica approach is provided to reduce the overhead required for protection. This will be of interest when the number of parallel filters is small and the cost of the proposed scheme is larger in that case. Another interesting topic to continue this technique is to search the use of more powerful multibit ECCs, such as Bose–Chaudhuri–Hocquenghem codes, to correct errors on multiple filters.

VI. REFERENCES

[1] M. Nicolaidis, "Design for soft error mitigation," *IEEE Trans. Device Mater. Rel.*, vol. 5, no. 3, pp. 405–418, Sep. 2005.

[2] A. Reddy and P. Banarjee "Algorithm-based fault detection for sig-nal processing applications," *IEEE Trans. Comput.*, vol. 39, no. 10, pp. 1304–1308, Oct. 1990.

[3] B. Shim and N. Shanbhag, "Energy-efficient soft error-tolerant digital signal processing," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 14, no. 4, pp. 336–348, Apr. 2006.

[4] T. Hitana and A. K. Deb, "Bridging concurrent and non-concurrent error detection in FIR filters," in *Proc. Norchip Conf.*, 2004, pp. 75–78.

[5] Christo Ananth, H.Anusuya Baby, "High Efficient Complex Parallelism for Cryptography", IOSR Journal of Computer Engineering (IOSR-JCE), Volume 16, Issue 2, Ver. III (Mar-Apr. 2014), PP 01-07

[6] S. Pontarelli, G. C. Cardarilli, M. Re, and A. Salsano, "Totally fault tolerant RNS based FIR filters," in *Proc. IEEE IOLTS*, Jul. 2008, pp. 192–194.

[7] Z. Gao, W. Yang, X. Chen, M. Zhao, and J. Wang, "Fault missing rate analysis of the arithmetic residue codes based fault-tolerant FIR filter design," in *Proc. IEEE IOLTS*, Jun. 2012, pp. 130–133.

[8] P. Reviriego, C. J. Bleakley, and J. A. Maestro, "Strutural DMR: A Technique for implementation of soft-error-tolerant FIR filters," *IEEE Trans. Circuits Syst., Exp. Briefs*, vol. 58, no. 8, pp. 512–516, Aug. 2011.

[9] P. P. Vaidyanathan. *Multirate Systems and Filter Banks*. Upper Saddle River, NJ, USA: Prentice-Hall, 1993.

[10] A. Sibille, C. Oestges, and A. Zanella, *MIMO: From Theory to Imple-mentation*. San Francisco, CA, USA: Academic Press, 2010.

[11] P. Reviriego, S. Pontarelli, C. Bleakley, and J. A. Maestro, "Area efficient concurren t error detection and correction for paral-lel filters," *IET Electron. Lett.*, vol. 48, no. 20, pp. 1258–1260, Sep. 2012.

[12] A. V. Oppenheim and R. W. Schafer, *Discrete Time Signal Processing*. Upper Saddle River, NJ, USA: Prentice-Hall 1999.

[13] S. Lin and D. J. Costello, *Error Control Coding,* 2nd ed. Englewood Cliffs, NJ, USA: Prentice-Hall. 2004.

[14] R. W. Hamming, "Error correcting and error detecting codes," *Bell Syst. Tech. J.*, vol. 29, pp. 147–160, Apr. 1950.