Gao, Z., et al. Fault-tolerant polyphase filters-based decimators for SRAM-based FPGA implementations. In: *IEEE Transactions on Emerging Topics in Computing, 10(2), April-June 2022, Pp. 591-601*

DOI: https://doi.org/10.1109/TETC.2021.3108556

Fault Tolerant Polyphase Filters-based Decimators for SRAM-based FPGA Implementations

Zhen GAO, *Member, IEEE*, Jinhua Zhu, Tong Yan, Anees Ullah, *Member, IEEE*, and Pedro Reviriego, *Senior Member, IEEE*

Abstract—To reduce the oversampling rate of baseband signals, decimation is widely used in digital communication systems. Polyphase filters (PPFs) can be used to efficiently implement decimators. SRAM-based FPGAs provide large amounts of resources combined with flexibility and are a popular option for the implementation of communication receivers. However, they are sensitive to soft errors, which limit their application in harsh environments, such as space. An initial reliability study on SRAM-based FPGA implemented decimation shows that the soft errors on around 5% of the critical bits in the configuration memory of the decimator would degrade the decimated signal dramatically. Based on this result, this paper proposes an efficient fault tolerance scheme, in which the high correlation between adjacent PPFs outputs is utilized to tolerate the fault of a single-phase filter, and a duplicate and comparison structure is used to protect the fault tolerance logic. Hardware implementation and fault injection experiments show that the proposed scheme can drastically reduce the number of critical bits that cause severe output degradation with 1.5x resource usage and 0.75x maximum frequency relative to the unprotected decimator. Therefore, the proposed scheme can be an alternative to Triple Modular Redundancy that more than triples the use of resources.

Index Terms—Decimation, Polyphase filters, Fault tolerance, Soft errors, SRAM-FPGAs.

1 Introduction

In digital communication systems, it is common that the baseband receiver is fed with a narrow band signal with high oversampling rate. One of the key reasons is to keep flexibility of the receiver. For example, in software radio platforms that support different types of input signals with different bandwidth (e.g. Universal Software Radio Peripherals (USRP) from NI [1]), the sampling rate at the output of the integrated Radio Frequency (RF) circuit is typically fixed to cover the widest target signal bandwidth. When lower bandwidth signals are used, decimation is performed to adapt the signal to the expected bandwidth in order to reduce complexity in the baseband processing [2],[3]. This idea is widely used in modern mobile networks where the base stations need to support different waveforms with different bandwidth, including 5G, LTE/LTE-Advanced for 4G, W-CDMA/ cdma2000/TD-SCDMA for 3G, and GSM/GPRS for 2G, and so on [4]. In addition, the oversampled narrow band signal is also usually used in the scenario of demultiplexing of multiple signals [5] or for accurate timing synchronization [6]. In all those scenarios, decimation is one of the processing modules needed to adapt to all possible signal bandwidths with high flexibility. Therefore, decimators are widely used in communication systems. Parallel Polyphase Filters (PPFs) are commonly used to implement decimation as they provide low complexity and can be implemented efficiently [3]. This is important for resource limited radio platforms, such as software defined radio platforms used in satellites [7], [8].

An important issue for circuits that are used in harsh environments (e.g. space) is reliability as they can suffer a number of errors due to, for example, radiation effects [9],[10]. This is the case for SRAM-based FPGAs that are a popular option for onboard processing due to their vast amount of computational resources and good reconfigurability [11],[12],[13]. In recent years, more and more Commercial Off the Shelf (COTS) SRAM-based FPGAs are used for space missions [14],[15]. These circuits are more sensitive to cosmic radiation, especially to the Single Event Upsets (SEUs) effect, which is one of the most frequent events [9],[10]. SRAM-based FPGAs can suffer two types of SEUs: errors on the configuration memory and errors on the user memory. Errors on the configuration memory can modify the implemented design and to remove them, re-configuration of the FPGA is needed. On the other hand, errors on the user memory elements (used to store parameters and data used by the design) do not modify the design but can corrupt the results. For the PPFs-based decimation implemented on SRAM-based FPGAs, the main part of the user memory consists of the filter coefficients, and for configuration memory consists of the multiplication and accumulation operation of filters and related routing between computation resources within the FPGA, e.g. lookup tables and

- *Z. Gao, J. Zhu, and T. Yan are with the School of Electrical and Information Engineering, Tianjin University, Tianjin 300072, China (e-mail: zgao@tju.edu.cn; zhujh@tju.edu.cn; Tyan@tju.edu.cn).*
- *A. Ullah is with University of Engineering and Technology, Peshawar, Abbottabad Campus, Abbottabad 220101, Pakistan (e-mail: aneesullah@uetpeshawar.edu.pk).*
- *P. Reviriego is with Universidad Carlos III de Madrid, 28911 Leganés, Spain (e-mail: revirieg@it.uc3m.es).*

DSPs. In [16], the impact of SEUs on these memories was evaluated, and the results showed that there is a need to protect PPFs-based decimators as some SEUs can have a large impact on the decimator´s output. Therefore, its FPGA-implementation needs to be protected if they are intended for applications that operate in harsh environments, like space.

As far as the authors know, there is no published research proposing protection schemes for PPFs-based decimation. As will be introduced later, the core structure of the PPFs-based decimator is a group of parallel filters. Coding-based fault tolerance schemes have been proposed to protect parallel filters with two typical patterns. First, for parallel filters with different inputs and same filter coefficients, redundant filters can be added and their inputs will be generated by coding. For example, [17] proposed to feed the redundant filters with the simple checksums of the inputs of the original filters, and [18] and [19] applied classical Hamming codes to generate the inputs of the redundant filters based on the inputs of the original filters. Furthermore, [20] applied real number coding to reduce the number of redundant filters. Second, [21] and [22] considered the parallel filters with the same input and different filter coefficients, and proposed to generate the coefficients of redundant filters with the combinations of those of original filters. In both cases, the precise mathematical relationship between the different inputs or filter coefficients is maintained among the parallel filter outputs due to the linear property of the filter processing, which could be used to identify the faulty filter and correct the wrong outputs. Unfortunately, the parallel filters in the PPFs-based decimator have both different inputs and different filter coefficients. Therefore, there is no way to introduce linear relationship between the outputs of PPFs by coding on their inputs or coefficients. In addition, the final output of the PPFs-based decimator is the sum of all the filter outputs, which could not be protected by existing coding-based schemes that protect only the parallel filters' individual outputs. A coding approach is also used in [23] for protection of adaptive filters, but only a single filter was considered in this work. In summary, existing works on the protection of parallel filters are not applicable to PPFs.

In this paper, an efficient protection scheme for PPFs-based decimators is proposed. The main novelty is to exploit the high correlation between the PPFs outputs to mitigate errors. This correlation is due to the operation of the parallel filters on subsets of samples and coefficients coming from the same stream of inputs and filter at the original sampling rate, respectively. Exploiting this algorithmic and structural property, we propose an efficient soft-error tolerant PPFs-based decimator by using adjacent filter outputs in place of faulty ones. The protection logic itself is protected with a novel Duplication with Comparison (DwC) scheme that can correct most errors. The overall proposed approach is mapped onto a Zynq SoC platform and evaluated with fault injection experiments in the configuration memory. Our results show that we are able to improve resilience to SEUs by 95% compared to an unprotected design at a cost of 1.5x the resource usage and 0.75x the maximum frequency.

The rest of the paper is organized as follows. In Section 2, the structure of PPFs-based decimators is introduced, and the reliability evaluation results are reported. Then an efficient fault tolerant scheme is proposed in Section 3 and evaluated in Section 4, respectively. Finally, the applicability of the proposed scheme is discussed in Section 5, and the paper is concluded in Section 6.

2 Reliability of PPFs-based Decimators

This section first describes the structure of PPFs based decimators. Then, the results of a reliability evaluation for an SRAM-based FPGA implementation presented in [16] are discussed. This provides the background needed for the proposed protection scheme presented in the next section.

2.1 Structure of PPFs-based Decimator

A direct way to reduce the sampling rate of a narrowband signal $x(n)$ by a factor of M is to filter the signal with a low pass filter (that attenuates frequencies above $1/M$) to remove the out-band noise, and then discard M-1 samples for each group of M samples. In the case of a Finite Impulse Response (FIR) filter $h(n)$, the *r-th* sample of the filtered signal can be expressed as:

$$y(r) = \sum_{n=0}^{N-1} h(n)\, x(r+n) \tag{1}$$

When using the above formula, all the *M-1* output samples that will be removed are also calculated, which is a huge waste of computational power, and this situation will get worse as the decimation factor increases. A better and more efficient scheme is to compute only those output samples that are actually used. In other words, if the decimation factor is two, every other output sample has to be calculated; if the decimation factor is four, every fourth output sample has to be calculated, and so on. The benefit of this method is that fewer computations are needed, which is more efficient and requires fewer hardware resources. Polyphase Filters (PPFs) are used to do such a processing for efficient decimation.

PPFs can be thought of as a composite filter with impulse response $h(n)$, where n = 0,1,2, ⋯, N – 1, that is segmented into M separate shorter length filters operating in parallel. Each of the M segments uses a subset of the filter coefficients in $h(n)$. Each segment is called a phase and is represented by $P_m(k)$ for k = 0,1,2, ⋯, K – 1, which can be expressed as in equation (2) for a filter with M phases. The number of composite coefficients N is chosen so that the number of coefficients per phase is an integer given by $K = N/M$.

$$P_m(k) = h(m + kM) \tag{2}$$

in which m = 0, 2, …, M-1 is the filter index, and k = 0, 1, 2, …, K-1 is the tap index. Therefore, the expression for the filtered and decimated output sequence $z(r)$ can be derived as:

$$z(r) = \sum_{m=0}^{M-1} y_m(r), \quad r = 0,1,...,L-1 \tag{3}$$

in which

$$y_m(r) = \sum_k P_m(k)\, x\big((r+k)M + m\big) \quad (4)$$

is the output of the m-th phase filter, and L is the length of the decimation results.

Based on equations (3) and (4), the structure of PPFs is shown in Fig. 1. For $1/M$ decimation, the input signal $x(n)$ is grouped into M streams with sampling rate of $1/M$, and each of them is filtered with a phase filter. Finally, the sum of the outputs of all PPFs produces the decimated signal $z(r)$. It is obvious that the main block within the dash frame is a parallel filtering structure, in which both the input signal and the filter coefficients are different for each phase filter. Since the inputs of the filters are de-multiplexed from one signal $x(n)$, and the coefficients of them are de-multiplexed from one filter $h(n)$, strong correlation exists among the outputs of different filters regardless of the specific inputs or filter coefficients. This correlation will be analyzed and used in the proposed fault-tolerant design presented in Section 3.

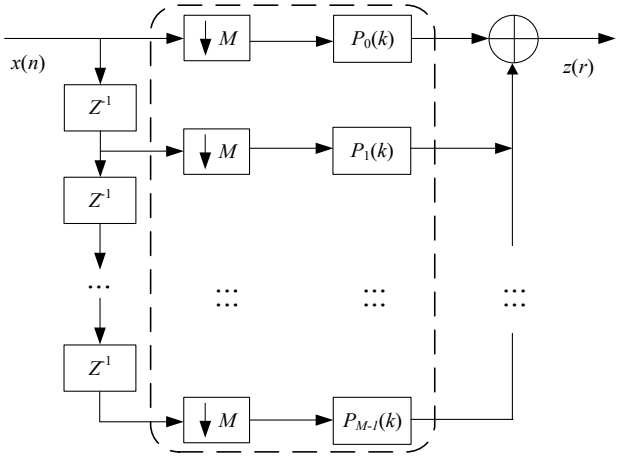

Fig. 1. Structure of PPFs-based $1/M$ Decimation

2.2 Reliability Evaluation of PPFs-based Decimator

The effect of SEUs on filter coefficients and configuration memory is evaluated based on the hardware-based fault injection experiments, which will be introduced in Section 4. In the experiments in this paper, the inputs to the decimator and the referenced decimation results are generated based on a Matlab platform as shown in Fig. 2, where '*os*' is short for oversampling rate. At the transmitter, Quad-Phase Shift Keying (QPSK) symbols are shape filtered and interpolated to samples with an oversampling rate of 64. Then, some noise is added to generate the input signal for the decimator with a Signal-to-Noise Ratio (SNR) of 10 dB. Finally, the reference decimated results without SEUs ($z^R(r)$) are generated according to the target output oversampling rate (e.g. *os* = 4 in Fig. 2). The total filter length is 64 in the experiments (N = 64), and the PPFs are generated according to equation (2), where the coefficients for the low-pass filter $h(n)$ are generated using the Matlab function *fir1* with cutoff frequency of $1/M$ (M = 16 in Fig. 2). Both the input signal and the filter coefficients are represented as 8-bit fixed point numbers. The effect of SEUs can be measured by the SNR_f (in dB) defined by equation (5), in which P_R is the power of $z^R(r)$, and $z^F(r)$ is the decimation results with SEUs.

$$SNR_f = 10 \times \lg\left(P_R \Big/ \left(\frac{1}{L} \sum_{r=0}^{L-1} \left| z^F(r) - z^R(r) \right|^2 \right) \right) \quad (5)$$

In addition, a Turbo code with 0.8 code rate is applied in the platform so that the effect of SEUs can also be assessed by the Bit Error Rate (BER). With the SNR loss due to synchronization errors (L_{syn}), the SNR at the input of the demodulator can be estimated as

$$SNR_D = -10 \times \lg\left(10^{-SNR/10} + 10^{-SNR_f/10}\right) - L_{syn}. \quad (6)$$

In the fault free case (SNR_f= +∞), we have zero BER with L_{syn} of 1dB. In following analysis, the BER with SEUs on the decimator is measured by feeding the decimation results from the corrupted decimator in the FPGA to the demodulator in the Matlab platform.

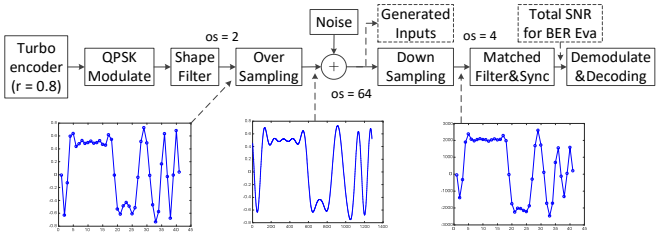

Fig. 2. Matlab simulation platform (target output *os* = 4, M = 16)

TABLE I. SNR_f for SEUs on filter coefficients

bit location	7	6	5	4	3	2	1	0
SNR_f (dB)	23.3	29.3	35.3	41.4	47.4	53.4	59.4	65.4

The filter coefficients are the main part of the user memory in PPFs-based decimators, and the effect of SEUs on the different bits of filter coefficients are shown in Table I for a decimation rate of 16 (output *os* = 4). The values in Table I are the averaged result over all 64 filter coefficients for 200 different inputs generated according to Fig. 2. As we can see from the table, SEUs on higher bits produce lower SNR_f, which is consistent with that observed for filters and other signal processing circuits [24], [25]. But even the SEUs on the highest bit (sign bit) would cause SNR_f larger than 23 dB. In addition, we notice that the difference of SNR_f for two adjacent bits is about 6 dB. This is reasonable because 1-bit higher means twice in amplitude, or 4 times in power. Therefore, the SEUs on the position that is 1-bit higher, would cause a 6 dB difference to the output signal.

As we have reported in [16], the length of the total filter coefficients has little influence on the effect of SEUs, but lower decimation rates could cause up to 3 dB SNR loss (*os* = 4 vs. *os* = 16). This means that the lowest SNR_f in practice is still over 20 dB. In this case, the SNR at the demodulator input is larger than 8.6 dB according to equation (6), and the BER is still lower than 10^{-8}, which is a negligible degradation to the system performance.

The same decimator (N = 64, M = 16) was implemented in HDL and synthesized in Vivado 2018 for a target FPGA. In the implementation, the bit-width for the PPFs output (y_m) and the final decimation result (z) are 18 and 22, respectively. After mapping the decimator on a Zynq-7020, we found that the number of configuration bits that are related to the decimation function (essential bits) is

1223766, and SEUs on 278641 of those bits would change the decimation results (critical bits). The Cumulative Distribution Function (CDF) statistics of the SNR_f of the corrupted outputs are plotted in Fig. 3. We can see that over 92% of the critical bits would cause a negligible degradation to the decimated signal (SNR_f > 20 dB), and the percentage of critical bits that cause SNR_f smaller than 10 dB is about 3%. For an SNR_f of 10 dB, the SNR at the input of the demodulator is about 6 dB, and the BER in this case is about 10^{-5}, which is an obvious performance degradation. Considering that the SNR loss due to synchronization errors (L_{syn}) would also be larger due to the signal distortion caused by SEUs, the actual BER would be higher than 10^{-5}. In the following analysis, we take SNR_f < 10 dB as the threshold for severe degradation, and define the critical bits causing SNR_f < 10 dB as the *severe bits*.

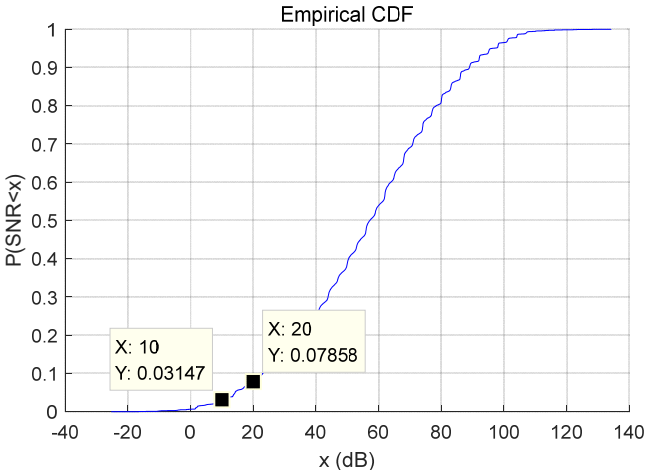

Fig. 3. CDF of SNR_f for unprotected decimator

The previous results reveal that the configuration memory bits that cause low SNR_f are the most critical ones and need protection. The use of Triple Modular Redundancy (TMR) or even Double Modular Redundancy (DMR) introduces a large cost in terms of FPGA resources and therefore, more efficient fault tolerance schemes are needed. Since algorithm-based fault tolerance is an effective way for efficient protection of digital signal processing circuits [26], [27] [28], the properties of PPFs-based decimators are explored in the next section and exploited to provide efficient fault tolerant designs.

3 Efficient Fault Tolerant Design

As introduced in Section 2, the PPFs-based decimator is composed of a group of filters and a final addition operation. This section will first verify that high correlation exists between the outputs of adjacent filters. This property is then combined with DMR of the final sum and a smart fault detection method to construct an efficient fault-tolerant design for the PPFs-based decimator.

3.1 Correlation Analysis between Adjacent Filter Outputs

For the outputs of two adjacent filters $y_m(r)$ and $y_{m+1}(r)$, r = 0, 1, …, L-1, the correlation between them can be measured as:

$$\rho(m,m+1)=\frac{1}{L-1}\sum_{r=0}^{L-1}\left(\frac{y_m(r)-u_m}{\sigma_m}\right)\left(\frac{y_{m+1}(r)-u_{m+1}}{\sigma_{m+1}}\right) \quad (6)$$

in which u and σ are the mean and standard deviation of the corresponding filter output. The correlation between adjacent filter outputs is evaluated based on the Matlab simulation platform in Fig. 2 in three aspects, which are input oversampling rate, total filter length and decimation rate. By averaging over 200 experiments with different inputs, the correlation coefficients are shown in Fig. 4. From the figure, we can conclude that larger decimation rates result in higher correlation between adjacent filter outputs, and the correlation is almost the same for a fixed decimation rate regardless of the total filter length and the input oversampling rate.

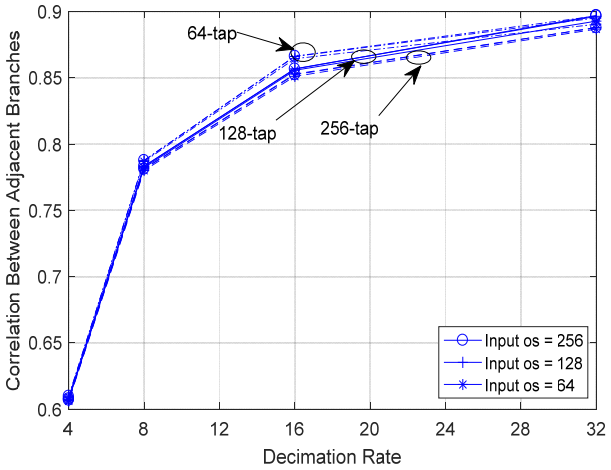

Fig. 4. Correlation between adjacent branches outputs in PPFs

Based on the high correlation between outputs of adjacent phase filters, if one filter in PPFs is corrupted by SEUs, it is reasonable to substitute its output with the average of the two adjacent filter outputs as shown in equation (7).

$$y_m^S(r)=\begin{cases} y_1(r), & m=0 \\ \left(y_{m-1}(r)+y_{m+1}(r)\right)/2, & 0<m<M-1 \\ y_{M-2}(r), & m=M-1 \end{cases} \quad (7)$$

In this case, some noise will be introduced into the decimation results, and the effect could be measured by SNR_s which is defined similarly to equation (5) except that $z^F(r)$ is replaced by $z^S(r)$ which is obtained by substituting one of the filter outputs $y_m(r)$ by $y_m^S(r)$. Fig. 5 shows the SNR_s for different decimation rates, input oversampling rates and total filter lengths. As we can see from the figure, the total filter length and the input oversampling rate have little influence in the SNR_s, which is reasonable because they do not affect the correlation between adjacent filter outputs. As expected, higher decimation rates would cause higher SNR_s, and the value is as high as 20 dB even for a decimation rate of 4.

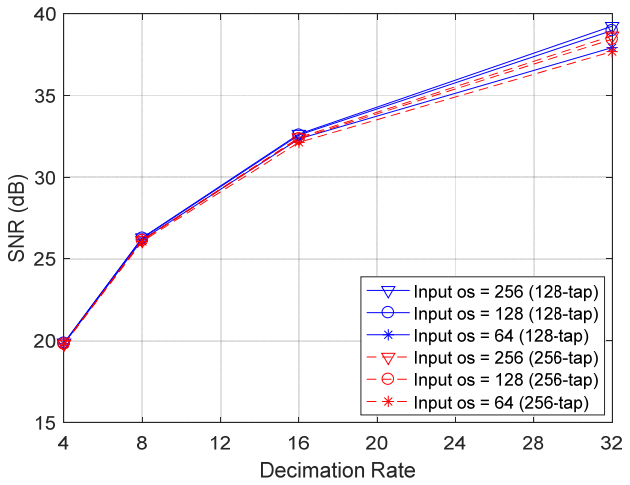

Fig. 5. SNR_s by substitution of the output of a single-phase filter

The above results verify that since a strong correlation exists between the outputs of adjacent phase filters, the substitution of the output of a single-phase filter with its adjacent ones would not introduce a large corruption in the decimation results. So, if SEUs corrupt a phase filter and it could be identified, its outputs could be recovered by the adjacent filter outputs.

3.2 Fault Tolerance Design for PPFs-based Decimators

As shown in Fig. 1, the PPFs-based decimator contains two parts, the PPFs part and the sum part. For the PPFs part that accounts for most of the complexity of the decimator, we make full use of the high correlation between phase filters to minimize the effect of a single filter failure on the decimation results. Then, the fault tolerance logic for PPFs is combined with the sum part and protected as a whole. The scheme is introduced next, presenting first its structure and then covering the details.

1) Overall structure of the fault tolerance design

The overall structure of the SEU tolerant PPFs-based decimator is shown in Fig. 6. The structure of the Sum & Fault Tolerance module is shown in Fig. 7, which is a Duplicate with Comparison (DwC) structure. In Fig. 7, each Sum & PPFs Protection module is used for the detection of a single faulty phase filter and the compensation of the error in the decimation results based on the high correlation between adjacent phase filters. Finally, the Compare & Select module is used for fault tolerance for the Sum & PPFs Protection logic based on the DwC structure.

2) Function of the Sum & PPFs Protection module

The functional structure of 'Sum & PPFs Protection' module is shown in Fig. 8. As we can see, the sum of the outputs from all the PPFs is denoted as z', and the mean value is calculated as a threshold ($y_{th} = z'/M$). The output of each phase filter y_m is compared with the threshold, and the result c_m would be 0 or 1, respectively, for the case that $y_m < y_{th}$ or $y_m > y_{th}$. In the fault free case, we would expect that approximately half of the PPFs outputs are larger than the mean value (y_{th}) and half of the PPFs outputs smaller than y_{th}. However, if a single-phase filter is corrupted by SEU, and causes large change of the final sum and thus of the threshold, its output would be very different from others. Based on this observation, the 'Fault Filter Detect' module works as follows:

(a) Calculates the sum of all comparison results as $C = \sum_{m=1}^{M} c_m$;

(b) If $C = M$-1 (M-1 1s and one 0 among all c_m) or $C = 1$ (M-1 0s and one 1 among all c_m), we can identify the faulty phase filter as the one that generated the single 0 or 1 (e.g. the m-th filter), and a correction of the output can be generated as $\Delta z = y_m^S - y_m$, where y_m^S is calculated according to equation (7). Then the final output is adjusted as $z = z' + \triangle z$ so that the contribution of y_m to z is replaced by y_m^S;

(c) Otherwise, all phase filters are correct, and $\triangle z = 0$.

In addition to the result z, the sum before correction (z') and the value of C are also provided to the Compare & Select module and are used for fault detection if one of the Sum & PPFs Protection modules is corrupted by an SEU.

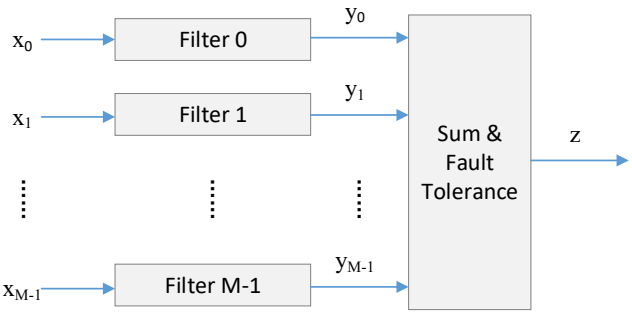

Fig. 6. Overall Structure of the SEU-tolerant PPFs-based decimator

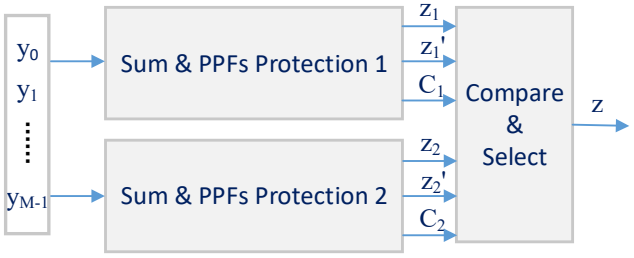

Fig. 7. DwC protection for the fault tolerance logic for PPFs

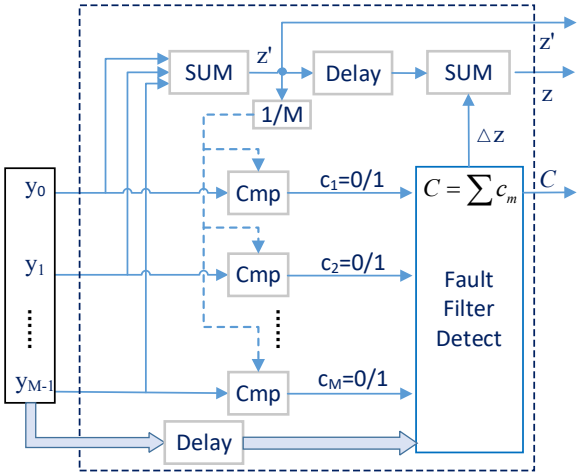

Fig. 8. Functional structure of 'Sum & PPFs Protection' module

3) Procedure of the Compare & Select module

After receiving the two groups of outputs from two Sum & PPFs Protection modules, the final Compare & Select module works according to the procedures shown in Fig. 9. The basic logic behind the scheme is as follows:

(a) If the two decimator outputs are the same, it means both Sum & PPFs Protection modules work well. Then any one of the decimator outputs can be chosen as the final output;

(b) If the two decimator outputs are different, it means that one of the Sum & PPFs Protection modules is faulty. Then the one that provides the C that is closer to M/2 is taken as the correct one, and the corresponding output is selected as the final decimation result;

(c) If the two decimator outputs are different but $C_1 = C_2$, it means that one of the decimator results is correct for mean calculation and comparison, but it was corrupted before output. Then the one with $z = z'$ is selected as the correct result.

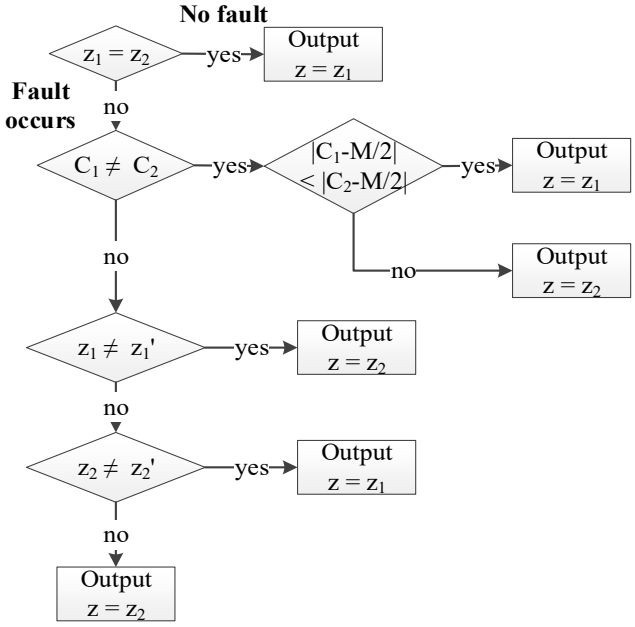

Fig. 9. Operation procedures of Compare & Select module

3.3 Reliability Analysis of the Proposed Fault Tolerance Scheme

With the assumption that only one part of the whole system may be corrupted by an SEU at a given time, the reliability of each functional unit in the proposed fault tolerant PPFs-based decimator is analyzed next. It should be noticed that a large part of the configuration bits in the FPGA implementation are used for wiring connections, and they are determined by the synthesis process. The reliability for these parts cannot be predicted in theory and can only be evaluated by hardware-based fault injection experiments.

1) For SEUs on PPFs

If one phase filter is corrupted by an SEU, its wrong outputs will change the final decimation result. If the wrong filter output is so different from the correct value that it is the only output over or below the mean value of all the filter outputs at that moment, it would be identified and corrected by the Sum & PPFs Protection module. Since both Sum & PPFs Protection modules work correctly, they will produce the same result ($z_1 = z_2$), then z_1 is selected as the final output.

It should be noticed that there is another hidden 'fault tolerance' idea behind for the case when a single-phase filter is corrupted by an SEU, but only a small change is introduced to the corresponding output so that it is still close to the outputs of all other phase filters. This would produce $0 < C < M$-1, and the 'tiny fault' would be ignored by the scheme. This is preferred because the small signal degradation would be comparable with that introduced by adjacent outputs substitution, so the subsequent fault tolerance processing would be meaningless and waste of power even if we could detect those small faults.

2) For SEUs on the Sum & PPFs Protection module

The effect of SEUs on each part of the Sum & PPFs Protection module (in Fig. 8) is analyzed in the following.

(a) If the first sum operation is corrupted so that z' is wrong, then the mean value would be wrong. In this case, the sum of the comparison results (C) would be away from $M/2$, so the comparison between C_1 and C_2 would identify the faulty Sum & PPFs Protection module.

(b) If the calculation of '$1/M$' is corrupted, the value of C would be wrong, and it would not be 1 or M-1 with high probability. In this case, the Fault Filter Detect logic would produce $\triangle z = 0$, so we would have $z_1 = z_2$, and the output would not be changed. There also exists a very small probability that the wrong mean value produces C = 1 or M-1. In this case, the fault could be identified by the comparison of C_1 and C_2.

(c) If one of the comparison operations in Fig. 8 is corrupted, the value of C would be changed by 1, which would not change the result of the Fault Filter Detect logic for most cases. For the extreme cases that C is changed from M-2 to M-1 or from 2 to 1, the fault could be tolerated by the comparison of C_1 and C_2.

(d) If the operation of $C = \sum_{m=1}^{M} c_m$ in the Fault Filter Detect module is corrupted, the probability that the wrong value of C is 1 or M-1 is very low. So, the fault would be ignored for most cases. For the very small probability that C happens to be 1 or M-1, the fault could be identified by the comparison between C_1 and C_2 in the final Compare & Select module.

(e) If the judgement based on the value of C is wrong, one of the phase filters is thought to be wrong by mistake, and an increment of $\Delta z = y_m^S - y_m$ would be used to adjust the result. In this case, the comparison between z and z' can identify the fault.

(f) Fault on the delay of sum result would produce a wrong output, which could be identified by the comparison between z and z' in the Compare & Select module.

(g) If the calculation of y_m^S or the delay for PPFs outputs get corrupted, the fault will be masked naturally because this logic would not be used for $0<C<M$-1.

3) For SEUs on Compare & Select module

The main body of the Compare & Select module is to output z_1 or z_2 for different conditions. If only this logic is corrupted, the decimation result would not be affected in theory because both of z_1 or z_2 are correct.

4 Performance Evaluation

This section evaluates the proposed scheme considering both the resource overheads introduced over an unprotected implementation and its effectiveness in avoiding large errors at the decimator output.

4.1 Resource Overhead

To evaluate the performance of the proposed protection technique, a 1/16 decimator (M = 16) with total filter length of 64 (N = 64) is used as a case study. The design has been implemented in HDL and mapped to a Zynq-7000 SoC (XC7Z020) hosted in a Zedboard. As a reference, the unprotected decimator was also implemented on the same board. Vivado 2018.2 is used for compile and synthesis, and the synthesis option of 'resource_sharing' is disabled to ensure that all modules are independent and isolated in all the implementations.

Based on the synthesis results in Vivado, the resource usage (Lookup Tables (LUTs), registers and BlockRAMs) and the maximum frequency for the unprotected and protected decimator are compared in Table II. From the results it can be observed that the overhead of protected decimator is around 1.5 times of the unprotected one, and two BlockRAMs are used in the protected decimator for delay of the outputs of PPFs. This overhead is significantly lower than that of Triple Modular Redundancy (TMR) that requires more than 3x of the resources of an unprotected design. The maximum frequency for the protected decimator is lower than that for the unprotected one by about 25%. However, considering that the maximum bandwidth is smaller than 100MHz in most mobile communication standards, the protected decimator that can support a sampling rate of 338Mhz can be applied in most practical systems. Finally, a latency of two cycles is introduced in the protected design by the delay of sum in Fig. 8 and the comparison in Fig. 9.

TABLE II. Overhead evaluation of the unprotected and protected decimator (N = 64, M = 16)

	LUTs	Registers	BRAM	Max Freq.
Unprotected	6603	4930	0	450MHz
Protected	9887 (1.5x)	7704 (1.56x)	2	338MHz (0.75x)

4.2 Platform for SEU Injection on User and Configuration Memory

To evaluate the reliability of the PPFs-based decimators to SEUs on the filter coefficients and configuration memory, an adapted version of the fault injection tool presented in [28] was used on a Zedboard as in [16]. As shown in Fig. 10, the fault injection platform consists of two parts, the Programmable System (PS) and the Programmable Logic (PL), and the two regions are connected through AXI buses. The PS consists of the ARM Cortex-A9 processor and the dedicated controllers for some peripherals e.g. DDR memory, SD card, and UART. The DDR memory is used to store the bit lists of configuration memory and filter coefficients that need to be upset, which are generated during the compile process in Vivado. The UART module is responsible for logging results to the PC. The PL part consists of two copies of the decimator i.e. Golden and Design Under Test (DUT). In addition, a comparator and a synchronizer block are also implemented in the PL part. The comparator is used to calculate the differences between the outputs from the Golden and DUT decimator and communicating the results to the ARM processor. The synchronizer is responsible for controlling the clock to DUT and Golden copies of decimators. The inputs to the decimators are generated by the Matlab simulation and fed by the ARM processor. Another important module in the PL part is the Internal Configuration Access Port (ICAP), which allows the ARM processor to access the configuration memory and the filter coefficients related to the DUT decimator and modify them in run-time for SEU injection.

The software running on the ARM processor controls the fault injection process. First, the clock to the decimators is frozen through the synchronizer. Then, one frame of the configuration memory or one filter coefficient is read back through the ICAP port. Based on the address of the configuration memory bits or coefficient bits in the fault lists, one bit is upset and the corrupted frame or coefficient is written back to the DUT. Finally, the clock is re-started and two outputs are generated by the two decimators for the same input from the ARM processor. The number and position of the critical bits for DUT are recorded in the ARM processor, and the corresponding decimation results are saved into separate files, which are post-processed by the Matlab platform for SNR_f and BER evaluation. The process is repeated for all essential bits and coefficient bits to characterize the reliability of the proposed fault-tolerant design.

4.3 Reliability Evaluation to of Protected Decimator to SEUs on Configuration Memory

To have a clear understanding of the reliability of the proposed fault tolerance scheme, each functional unit of the protected PPFs-based decimator is tested separately on the fault injection platform, and the number of essential bits, critical bits and severe bits for each part are shown in Table III, including the ratio of critical bits over essential bits and that of severe bits over critical bits. For convenience of comparison, the values for the unprotected decimator are also included at the bottom line of the table. As we can see, the absolute number of critical bits for the protected decimator is larger than that of the unprotected one, and the PPFs contribute around 75% of that number. The percentage of critical bits over essential bits for the fault tolerance logic is relatively much smaller, which is due to the logic added for correction being less critical than the one in the decimator itself which seems reasonable.

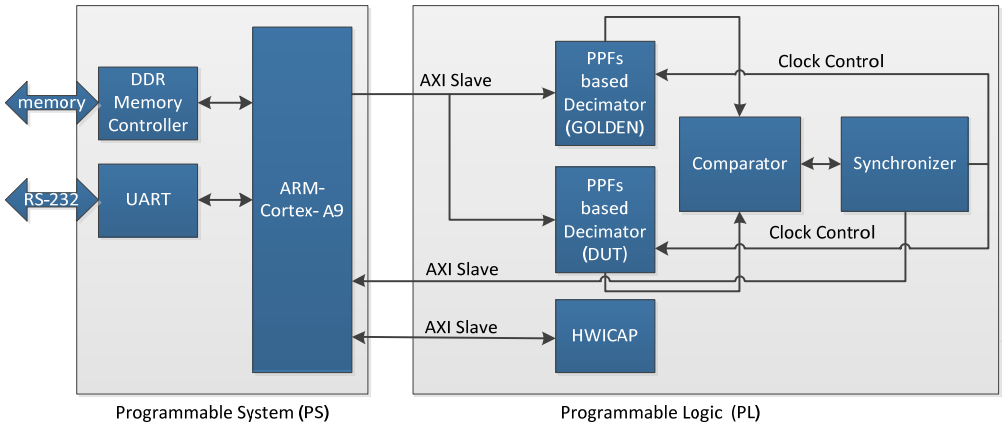

Fig. 10. Zynq-SoC-based fault injection platform (from [16])

Table III. Essential, critical and severe bits for protected and unprotected PPFs-based decimators

	Essential bits	Critical bits	Severe bits	Critical / Essential	Severe / Critical
PPFs	1139806	235940	0	20.7%	0
Sum & PPFs Protection	1048172	65416	0	6.24%	0
Compare & Select	56198	3000	409	5.34%	13.6%
Total of Protected	2244176	304356	409	13.56%	0.13%
Unprotected	1223766	278641	8769	22.77%	3.14%

However, the key objective is not to reduce the absolute number of critical bits of the decimator, but to reduce the number of severe bits causing SNR_f < 10 dB. To show the effectiveness of the proposed scheme, the CDFs of SNR_f for each part of the decimator are plotted in Fig. 11. As we can see from the figure, the critical bits from the main parts (235940 for PPFs and 65416 for Sum & PPFs Protection module) will not cause SNR_f smaller than 10 dB, which proves that the SEUs that could cause low SNR_f are tolerated successfully. The final Compare & Select module is not under protection, so SEUs on about 13.6% of the critical bits would cause SNR_f < 10 dB (as marked in Fig. 11). However, since there are only 3000 critical bits in the final module, the absolute number is 409, which is only 4.7% of that for unprotected decimator (409/8769 = 0.0466). In other words, the number of critical bits that may cause severe performance degradation (BER > 10^{-5}) is reduced by 95%. Further analysis shows that most of these critical bits are the ones for the signal connections between Sum & PPFs Protection modules and the Compare & Select module, and the ones for final output signals. Such SEUs will introduce a continuous error in the output signal, which causes low SNR_f.

Fig. 12 compares the CDFs of SNR_f for the unprotected and protected decimators, in which the percentage of severe bits causing large BER are marked for the unprotected decimator. As we can see, in addition to mitigating SEUs that cause low SNR_f, the curve of the protected decimator is also shifted to the right by about 2 ~ 5 dB relative to that of the unprotected decimator. The PDFs of SNR_f for the unprotected and protected decimators in Fig. 13 reveal more insights. It is obvious that SEUs that cause SNR_f lower than 10 dB for the unprotected decimator are replaced by the ones that cause SNR_f around 40 dB for the protected decimator. This corresponds to the cases that the SEUs on severe bits for one phase filter are detected, and the outputs of adjacent filters are used to compensate the errors introduced by the faulty phase filter in the decimation results. Based on Fig. 5, this compensation will introduce an SNR_s over 33dB for M = 16. The cost paid for this improvement is a 1.5x resource usage and a 0.75x maximum frequency due to the fault tolerance for PPFs and the DwC based protection of the fault tolerance logic.

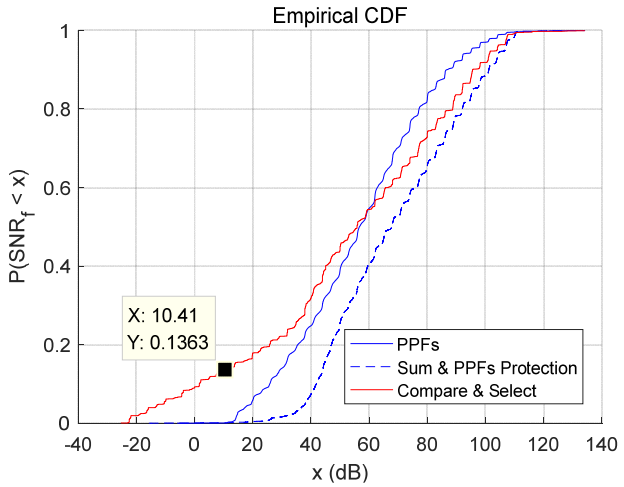

Fig. 11. CDF of SNR_f of critical bits for each function unit

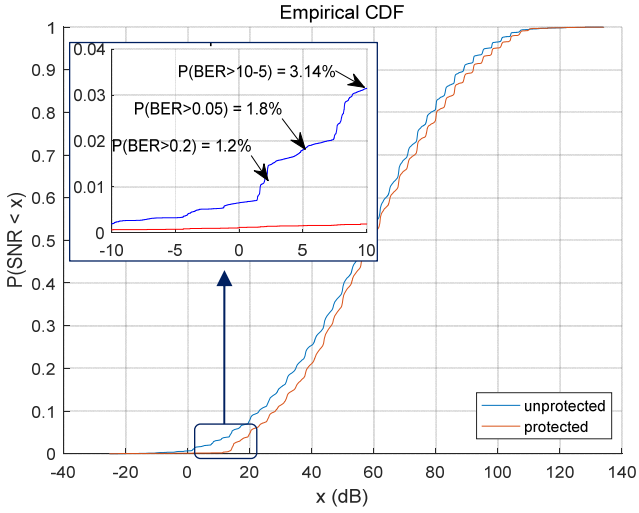

Fig. 12. CDF of SNR_f for protected and unprotected decimator

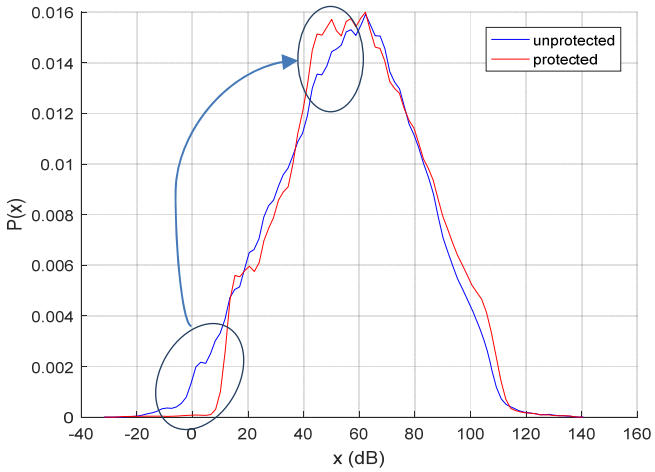

Fig. 13. PDF of SNR_f for unprotected and protected decimator

5 Discussion and Practical Considerations

Results in Section 4 illustrate how the proposed technique is able to minimize the impact of SEUs on the SNR and thus on the system performance (BER). In this section, we will first discuss the performance of the proposed scheme for small decimation rates, and then the application of the scheme in practical systems.

As reported in [16], if the decimation rate decreases from 16 to 4, the portion of severe bits will increase by about 10%. Although the correlation between adjacent phase filters decreases for small decimation rates (as shown in Fig. 4), the SNR_s is still as large as 20 dB for M = 4 according to Fig. 5. So more severe bits from the PPFs part will be tolerated by the proposed scheme. Additionally, the severe bits of the proposed scheme mainly come from the final Compare & Select module, which is not related to the decimation rate. Considering all these factors, the reliability improvement for small decimation rates would be similar to that for large decimation rates.

The proposed scheme is designed to tolerate a single soft error on the PPFs-based decimator. Therefore, in practice, periodic scrubbing is needed to correct the single soft errors so that accumulation of soft errors is avoided. Since the scrubbing interval can be as large as tens of milliseconds, e.g. ~8 ms for XC7Z020 and ~27 ms for XC7Z045 at the maximum ICAP operation frequency [30], we may have a transient SNR drop (slight for most cases) for the transmission of several packets. For example, for a signal with symbol rate of 5MHz, a maximum of $5x10^4$ symbols may experience an SNR degradation when scrubbing is done each 10 ms. In this paper, the influence of the SNR drop is proved to be negligible for a system with QPSK and 4/5 Turbo codes. For higher levels of modulation (e.g. 16 QAM or 64 QAM) with larger decoding thresholds, adaptive modulation and coding (AMC) could be used to adapt to *the transient degradation of the channel*, e.g. switching to QPSK. In this case, the SEU would only cause an acceptable throughput reduction without transmission interruptions. Although the AMC could be also used for the case of the unprotected decimator, the large SNR drop will introduce severe transmission throughput reduction or even transmission interruption such that no adaption scheme can work under the *severe channel degradation*.

6 Conclusion

In this paper, the fault tolerance of Polyphase Filters (PPFs) based decimators implemented on SRAM-FPGAs has been considered. The initial reliability study by fault injection of the PPFs based decimator implemented on SRAM-FPGAs shows that SEUs on 5% of the critical bits of the configuration memory would cause severe degradation of the decimation result. Therefore, to ensure a reliable operation, PPFs-based decimators need to be protected. However, the cost of the traditional TMR or DMR fault tolerance schemes is large and may be not acceptable in some systems. This means that more efficient protection schemes are needed. For this purpose, we studied the relationship between the filters in the PPFs and found high correlation among the multiple output streams. This property is utilized for faulty phase filter detection and correction based on the substitution by adjacent filter outputs. Finally, a duplicate and comparison structure is constructed to protect the fault tolerance logic itself. Hardware implementation and fault injection experiments show that the number of critical bits that cause severe output degradation could be decreased by 95% at a cost of 1.5x resource usage and 0.75x maximum frequency. These results prove the efficiency and effectiveness of the proposed scheme.

Acknowledgment

This work is supported by Natural Science Funds of China (Grant No. 62171313) and is partially supported by the ACHILLES project PID2019-104207RB-I00 funded by the Spanish Ministry of Science and Innovation and by the Madrid Community research project TAPIR-CM grant no. P2018/TCS-4496.

References

[1] Ettus Research, "USRP Hardware Driver and USRP Manual," Version: 3.14.0.0.rc1-115-gb2dd1655d, available online: http://files.ettus.com/ manual/page_usrp_e3x0.html.

[2] R. E. Crochiere and L. R. Rabiner, *Mutltirate Digital Signal Processing*, Englewood Cliffs, NJ: Prentice-Hall, 1983.

[3] P. P. Vaidyanathan, *Multirate systems and filter banks*, Englewood Cliffs, NJ: Prentice-Hall, 1993.

[4] S. Zhou, T. Zhao, Z. Niu, et al., "Software-defined hypercellular architecture for green and elastic wireless access," in *IEEE Communications Magazine*, vol. 54, no. 1, pp. 12-19, January 2016.

[5] F. J. Harris and T. Weschselberger, "Multirate all-digital modems for transport of universal multiplex transport layer for digital compression," in *Proceeding of the National Cable & Tele-*

communications Association Conference, San Francisco, CA, June 1993.

[6] F. J. Harris and M. Rice, "Multirate digital filters for symbol timing synchronization in software defined radios," in *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 12, pp. 2346-2357, Dec. 2001.

[7] BIS Research, "Global Software-Defined Satellite Market: Focus on End User, Mass, Orbit Technology, Subsystem, and Services - Analysis and Forecast, 2019-2030," Oct. 2019.

[8] S. Xu, X. Wang and M. Huang, "Software-Defined Next-Generation Satellite Networks: Architecture, Challenges, and Solutions," in *IEEE Access*, vol. 6, pp. 4027-4041, 2018.

[9] F. L. Kastensmidt, L. Carro L and R. Reis, Fault-tolerance Techniques for SRAM-based FPGAs, New Haven, Springer 2006.

[10] N. Kanekawa, E. H. Ibe, T. Suga, et al., Dependability in electronic systems: mitigation of hardware failures, soft errors, and electro- magnetic disturbances, Springer Verlag, New York, USA, 2010.

[11] S. Lopez, T. Vladimirova, C. Gonzalez, et al., "The Promise of Reconfigurable Computing for Hyperspectral Imaging Onboard Systems: A Review and Trends," in *Proceedings of the IEEE*, vol. 101, no. 3, pp. 698-722, March 2013.

[12] F. Siegle, T. Vladimirova , J. Ilstad, et al., "Availability analysis for satellite data processing systems based on SRAM FPGAs," in *IEEE Transactions on Aerospace and Electronic Systems*, vol. 52, no. 3, pp. 977-989, June 2016.

[13] D. Bekker, P. Pingree and T. Werne, "The COVE Payload - A Reconfigurable FPGA-Based Processor for CubeSats," in *Proceeding of the AIAA/USU Conference on Small Satellites*, 2011.

[14] R. Baumann, "From COTS to space grade electronics-improving reliability for Harsh environments," *2014 IEEE International Integrated Reliability Workshop (IIRW)*, South Lake Tahoe, CA, 2014.

[15] D. Stone, A. Lindenmoyer, G. French, et al. "NASA's approach to commercial cargo and crew transportation." *Acta Astronautica*, vol. 63, no. 1, pp. 192–197, July 2008.

[16] Z. Gao, J. Zhu, L. Yan, et al., "Reliability Evaluation of Polyphase-filter based Decimators Implemented on SRAM-FPGAs," *2019 IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)*, Noordwijk, Netherlands, 2019, pp. 1-4.

[17] W. Song and B. R. Musicus, "A fault-tolerant architecture for a parallel digital signal processing machine," in *Proceeding of IEEE International Conference on Computer Design*, pp. 385-390, 1987.

[18] Z. Gao, P. Reviriego, W. Pan, et al., "Fault Tolerant Parallel Filters Based on Error Correction Codes," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 23, no. 2, pp. 384-387, Feb. 2015.

[19] M. Kasaiah and K. Babulu, "Comparative performance analysis of FIR and IIR filters based on error correction codes," *International Conf. on Communication and Electronics Systems*, Coimbatore, 2016.

[20] Z. Gao, P. Reviriego, Z. Xu, et al., "Efficient Coding Schemes for Fault-Tolerant Parallel Filters," in *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 62, no. 7, pp. 666-670, July 2015.

[21] Y. Yang, P. Grover and S. Kar, "Fault-tolerant parallel linear filtering using compressive sensing," *9th International Symposium on Turbo Codes and Iterative Information Processing (ISTC)*, Brest, 2016.

[22] Z. Gao, M. Zhou, P. Reviriego, et al., "Efficient Fault-Tolerant Design for Parallel Matched Filters," in IEEE Transactions on Circuits and Systems II: Express Briefs, vol. 65, no. 3, pp. 366-370, March 2018.

[23] S. Liu, G. Sorrenti, P. Reviriego, *et al.*, "Increasing Reliability of FPGA-Based Adaptive Equalizers in the Presence of Single Event Upsets," in *IEEE Transactions on Nuclear Science*, vol. 58, no. 3, pp. 1072-1077, June 2011.

[24] S. Byonghyo, S. R. Sridhara, and N. R. Shanbhag, "Reliable low-power digital signal processing via reduced precision redundancy," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 12, no. 5, pp. 497-510, May 2004.

[25] K. Chen, L. Chen, P. Reviriego, et al., "Efficient Implementations of Reduced Precision Redundancy Multiply and Accumulate," in *IEEE Transactions on Computers*, vol. 68, no. 5, pp. 784-790, 1 May 2019.

[26] A. L. N. Reddy and P. Banerjee, "Algorithm-based fault detection for signal processing applications," in *IEEE Transactions on Computers*, vol. 39, no. 10, pp. 1304-1308, Oct. 1990.

[27] A. Roy-Chowdhury, N. Bellas and P. Banerjee, "Algorithm-based error-detection schemes for iterative solution of partial differential equations," in *IEEE Transactions on Computers*, vol. 45, no. 4, pp. 394-407, April 1996.

[28] A. Roy-Chowdhury and P. Banerjee, "Algorithm-based fault location and recovery for matrix computations on multiprocessor systems," in *IEEE Transactions on Computers*, vol. 45, no. 11, pp. 1239-1247, Nov. 1996.

[29] A. Ullah A, E. Sanchez E, L. Sterpone, *et al.*, "An FPGA based dynamically reconfigurable platform for emulation of permanent faults in ASICs," *Microelectronics Reliability*, vol. 75, pp. 110-120, Aug. 2017.

[30] Xilinx, Soft Error Mitigation Controller v4.1, PG036 November 19, 2014

GAO Zhen (M'11)**,** received the BS, MS and PhD degree in Electrical and Information Engineering from Tianjin University, China, in 2005, 2007 and 2011, respectively. During 2008.10 ~ 2010.11, he was a visiting scholar in GeorgiaTech, working on the design and implementation for OFDM based cooperative communi- cations. During 2011.7 ~ 2014.12, he was a Postdoc researcher in the Wireless and Mobile Communication Research Center in Tsinghua University, China, working on mobile satellite communication and fault-tolerant design for DSPs. Since 2014.12, he is an Associate Professor in Tianjin University. His focus now includes fault-tolerant DSPs design and Blockchain technologies.

Zhu Jinhua, received the BS degree from Hunan Normal University in July 2017, received the MS degree from Tianjin University in Jan. 2020, and is a PhD student in Tianjin University since Sep. 2020. Her research focus now is radiation effects study and fault-tolerant design for digital signal processing.

Yan Tong, received the BS degree from Changchun University of Science and Technology in July 2018, and is a Master student in Tianjin University since Sep. 2018. His focus now is design and implementation of SRAM-FPGA based fault injection platform and radiation effects study for digital signal processing.

Anees Ullah, received B.Sc. and M.Sc. Degrees in Electrical Engineering for University of Engineering and Technology, Peshawar in 2009 and 2011, respectively. He received a PhD degree in Computer Engineering from Politecnico di Torino, Italy in 2015. During 2016-2017, he worked as a Post-doc researcher in Universidad Antonio de Nebrija, Spain. Currently, he is working as Assistant Professor in department of Electronics Engineering, University of Engineering and Technology, Peshawar. His research interests include fault-tolerant digital systems design, fault injection in FPGAs, FPGA-based Ternary Content Addressable Memories (TCAMs) and Approximate Computing.

Reviriego Pedro (SM'15), received the M.Sc. and Ph.D. degrees in telecommunications engineering from the Technical University of Madrid, Madrid, Spain, in 1994 and 1997, respectively. From 1997 to 2000, he was an R&D Engineer with Teldat, Madrid, working on router implementation. In 2000, he joined Massana to work on the development of 1000BaseT transceivers. From 2004 to 2007, he was a Distinguished Member of Technical Staff with the LSI Corporation, working on the development of Ethernet transceivers. From 2007 to 2018 he was with Universidad Antonio de Nebrija. He is currently with the Departamento de Ingeniería Telemática, Universidad Carlos III de Madrid.