Hindawi Publishing Corporation
International Journal of Reconfigurable Computing
Volume 2011, Article ID 897189, 12 pages
doi:10.1155/2011/897189

Research Article

Reduced-Precision Redundancy on FPGAs

Brian Pratt, Megan Fuller, and Michael Wirthlin

NSF Center for High-Performance Reconfigurable Computing (CHREC), Department of Electrical and Computer Engineering, Brigham Young University, Provo, UT 84602, USA

Correspondence should be addressed to Brian Pratt, brianpratt@byu.net

Received 20 May 2011; Revised 29 July 2011; Accepted 29 July 2011

Academic Editor: Salvatore Pontarelli

Reduced-precision redundancy (RPR) has been shown to be a viable alternative to triple modular redundancy (TMR) for digital circuits. This paper builds on previous research by offering a detailed analysis of the implementation of RPR on FPGAs to improve reliability in soft error environments. Example implementations and fault injection experiments demonstrate the cost and benefits of RPR, showing how RPR can be used to improve the failure rate by up to 200 times over an unmitigated system at costs less than half that of TMR. A novel method is also presented for improving the error-masking ability of RPR by up to 5 times at no additional hardware cost under certain conditions. This research shows RPR to be a very flexible soft error mitigation technique and offers insight into its application on FPGAs.

1. Introduction

Field-programmable gate arrays (FPGAs) are an attractive target for high-performance digital signal processing and real-time communication systems [1]. FPGAs have been used to implement communication-specific processors for well over a decade. Their ability to combine flexibility with good performance makes FPGAs popular for software-defined radios. Reconfigurable radios are also becoming more attractive for space-based applications. The ability to reconfigure the FPGA resources with an updated radio configuration reduces the amount of hardware needed on the spacecraft [2]. FPGAs are increasingly used in space for reconfigurable radios and other high-performance computing tasks [3–5].

The problem with using the popular SRAM- (static-random-access-memory-) based FPGAs in space is the presence of high-energy particles that may alter the operation of the digital circuitry or the state of static memory cells. These errors, called soft errors, do not cause any physical damage to the device but interact with state of memories or other digital circuits [6]. For example, charged particles can occasionally invert the contents of a memory cell. Such an event is called a "single event upset" (SEU) [7].

Because most of the FPGA area is devoted to static memory cells to store the FPGA configuration memory, FPGAs are very sensitive to radiation. Any FPGA design operating in space must consider the effects of high-energy radiation and implement some form of SEU mitigation. Triple modular redundancy (TMR) is the most popular SEU mitigation technique for FPGAs. TMR protects the FPGA circuit by creating three copies of a circuit and choosing the output based on a majority vote between the three. TMR masks the effects of SEUs as well as the less critical transient and soft data errors.

Although TMR is very effective at protecting FPGA circuits from soft errors, it is costly in terms of the circuit area, power, and circuit timing [8, 9]. A less expensive hardware mitigation strategy for arithmetic circuits is a technique called reduced-precision redundancy (RPR). RPR is designed to protect against large magnitude errors in arithmetic circuits by providing redundant, lower precision arithmetic circuits and comparing their results (the details of RPR will be described in Section 3). Although the use of RPR may introduce low precision errors, its area savings make it an attractive alternative for protecting FPGA signal processing circuits against SEUs, transient, and soft data errors.

RPR is a relatively new technique and is more difficult to implement than TMR. There are a number of important design decisions that must be made for each circuit protected by RPR. These choices include selecting the precision of the reduced-precision circuits and determining the threshold for detecting low-magnitude errors. This paper expands on previous work by clarifying the design space of these design choices and defining the trade-offs associated with these parameters (threshold selection is described in Section 5 and bit-width selection is described in Section 6). By understanding the impact of these design choices, more efficient SEU mitigation can be achieved. Using this insight, this paper introduces a new method to increase the effectiveness of RPR by up to 5 times for some systems with no additional hardware cost. The benefits of these techniques are validated on a matched filter for a binary pulse amplitude modulation (PAM) communication system. Using a well-proven fault injection technique, these experiments demonstrate significant hardware savings of RPR over TMR and acceptable levels of SEU mitigation.

2. Previous Work

RPR was introduced by Shim et al. as part of a power reduction technique for ASIC- (application-specific-integrated-circuit-) based DSP systems [10, 11]. Shim et al. used RPR to overcome errors introduced by voltage overscaling, which reduces the supply voltage of a circuit to save power. This voltage reduction slows the operation of the circuit and can cause intermittent errors at the circuit output when the longer logic paths are excited. RPR was used to reduce the effects of these intermittent errors, which had the tendency to occur in the most significant bits of the circuit output since those generally correspond to the longer paths through the logic.

Shim and Shanbhag later modified this RPR technique and analyzed it as a means for protecting against deep submicron noise and soft errors in ASIC-based DSP systems [12]. Reviriego et al. used this modification of RPR to protect an adaptive equalizer circuit in an ASIC system and took advantage of that circuit's error-correcting properties to reduce the cost of mitigation even further [13]. This soft error style of RPR is more suited towards SEU mitigation for FPGAs than the original. In a radiation environment, SEUs are distributed uniformly across an FPGA similar to soft errors in ASIC systems. These errors are not biased towards the most significant bits as in the VOS case. Still, because SEUs may impact the logic implemented by the FPGA, soft errors in ASIC systems tend to be less severe than those of concern in FPGAs.

Snodgrass presented an alternate RPR configuration and demonstrated it on FPGAs in [14]. Sullivan later provided details on how to implement this type of RPR on several elementary arithmetic operations and characterized the performance of some RPR systems in simulation [15]. Both of these authors confirmed that RPR could be a valuable SEU mitigation technique for certain FPGA-based systems.

Previous work demonstrated the viability of RPR for numerical systems. To use RPR in practice requires the designer to make a number of important design decisions such as the precision of the reduced-precision replicas used and the threshold for determining which of the three RPR modules, if any, is in error. This work extends the previous work by providing tools for making these choices, which trade off the area cost and performance of the RPR implementation. This paper offers more insight into the implementation of RPR on FPGA systems by introducing and discussing these trade-offs as well as providing detailed experimental results for systems using varying parameters. In addition, this paper suggests a novel experimental method for improving the performance of RPR with no additional hardware cost for certain systems.

3. Reduced-Precision Redundancy

RPR is implemented by creating two *identical* reduced-precision (RP) versions of the module to be protected, as illustrated in Figure 1. The outputs of the two RP modules are used to determine if there is a fault in the full-precision (FP) module. If the FP output differs from the RP outputs by *more* than a preset threshold, T_h, the FP module is assumed to be in error. When the FP module is found to be in error, the output of the RP modules is used instead as an estimate of the FP output. If the FP output differs from the RP outputs by *less* than T_h, the FP module is assumed to be correct and its output is used.

The arithmetic circuits protected by RPR may be of any size or complexity. The circuit may be an elementary arithmetic operation such as an adder or a more complex combination of operators such as a finite impulse response (FIR) filter (the effects of the size and complexity of the module to be protected on the efficiency and effectiveness of RPR are discussed in [15, 16]). This paper refers to the combination of full-precision and reduced-precision modules along with the decision hardware as an *RPR system* or *RPR module*.

Implementing RPR on a module requires the choice of two main parameters: the bit width of the reduced-precision module (B_r) and the decision threshold (T_h). The two values are linked and together greatly affect the cost and performance of RPR. The following two subsections introduce these parameters while Sections 5 and 6 describe the trade-offs for selecting the values of T_h and B_r in more detail.

3.1. RPR Bit Widths. The bit widths of the signals operated on in the RPR system have a great effect on both the cost and performance of the system. In Figure 1, the full n-bit input is truncated or rounded to a k-bit value (where $k < n$) before being passed to each of the RP modules. With a lower-precision input, the RP module can be made smaller than the FP module, reducing the cost of RPR compared to TMR. The lower the precision, however, the poorer the estimate the RP module offers of the FP result. This affects the ability of RPR to mask errors in the system. Section 6 describes these trade-offs in detail.

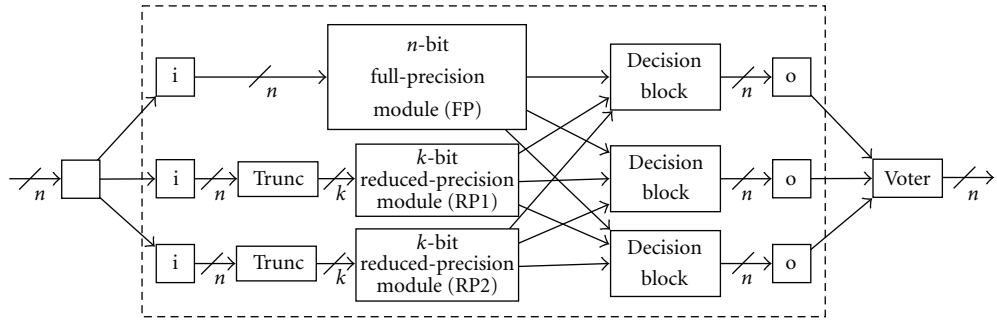

FIGURE 1: Simplified block diagram of an n-bit ($B = n$) full-precision module protected with RPR using two k-bit ($B_r = k$) reduced-precision modules, where $k < n$.

This paper refers to the bit widths of the full-precision and reduced-precision modules as B and B_r, respectively. For consistency, the numbers represented in this paper are in fixed point, twos complement format in the range $[-1, 1)$ with B/B_r bits to the right of the binary point and only the sign bit to the left. Thus the full-precision module has $B + 1$ input bits, and the reduced-precision modules have $B_r + 1$ input bits.

3.2. Decision Block. As shown in Figure 1, a decision block is included in RPR to determine if an error has occurred. Like TMR, RPR assumes that no more than a single upset occurs at one time. The decision block compares the outputs of the full-precision (FP) and two reduced-precision (RP1 and RP2) modules as follows:

if $((|\mathrm{FP}_{\mathrm{out}} - \mathrm{RP1}_{\mathrm{out}}| > T_h)$ AND $(\mathrm{RP1}_{\mathrm{out}} = \mathrm{RP2}_{\mathrm{out}}))$ **then**
  output $\Leftarrow \mathrm{RP2}_{\mathrm{out}}$
else
  output $\Leftarrow \mathrm{FP}_{\mathrm{out}}$
end if.

Thus the full-precision output is used when no error is found or when the two reduced-precision modules disagree. When the reduced-precision modules disagree, one of these must be in error rather than the full-precision module. Otherwise, one of the reduced-precision outputs is used, providing an estimate of the correct full-precision output.

For a particular instantiation of RPR (i.e., for a particular module and B_r value), there is an optimal range for T_h. If T_h is too large, the full-precision output will be used even when there are significant errors in that module. A T_h that is too small will cause the RP output to be chosen even when there are no errors in the FP module, resulting in the false detection (FD) upset case. The limits on the optimal range of T_h will be discussed in Section 5.

3.3. RPR Output Noise. The performance of an RPR system in the presence of soft errors can be measured by the deviation of its output from the unmitigated system in the absence of soft errors. In the context of DSP systems, this deviation could be termed "noise." The performance of an RPR DSP system, then, can be described in terms of the noise of the system in the presence of upsets.

Each individual upset causes a different amount of noise to be added to the system output. The amount of noise added to the output depends on the location of the upset within the circuit. For example, an upset affecting a high-order bit of computation is expected to cause more noise than an upset affecting a low-order bit.

Several noise signals and values are important in defining the operation and performance of RPR. The noise signal at the output of the RPR system is defined as difference between RPR system and the output of the full-precision module in the absence of upsets (the *true* output)

$$\epsilon_{\mathrm{RPR}} = \mathrm{FP}_{\mathrm{true}} - \mathrm{RPR}_{\mathrm{out}}. \tag{1}$$

The noise signal added by an upset in the full-precision module is the *upset error*

$$\epsilon_u = \mathrm{FP}_{\mathrm{out}} - \mathrm{FP}_{\mathrm{true}}. \tag{2}$$

The difference between the full-precision and reduced precision outputs is the *estimation error* signal

$$\epsilon_e = \mathrm{FP}_{\mathrm{out}} - \mathrm{RP}_{\mathrm{out}}. \tag{3}$$

And the value of the maximum estimation error is

$$\epsilon_{\max} = \max|\epsilon_e| = |\mathrm{FP}_{\mathrm{out}} - \mathrm{RP}_{\mathrm{out}}|. \tag{4}$$

3.4. RPR Upset Cases. The upsets in a system protected with RPR can be categorized by the location of the upset and its effect on the system. There are four possible upset cases for RPR in general.

(i) *Detected Upset (DU)*. An upset occurs in the full-precision module and the RPR decision block determines that there is an error in the full-precision module.

(ii) *Undetected Upset (UU)*. An upset occurs in the full-precision module but the RPR decision block does not indicate an error because the error does not exceed the threshold.

(iii) *False Detection (FD)*. Though there is no upset in the full-precision module, the RPR decision block indicates that there is an error. This could occur if

the natural difference between the full-precision and reduced-precision outputs (ϵ_e) is greater than the chosen value of T_h for some set of inputs, that is, $T_h < \epsilon_{\max}$.

(iv) *No Upset (NU)*. No upset exists in the full-precision module and there is no false detection.

The DU and the FD upset cases result in the RPR system choosing the reduced-precision output, operating in *reduced-precision mode*. The details of the RPR implementation control the distribution of upsets between the upset cases.

Each upset case has a distinct probability of occurrence and a distinct noise level or range that is added to the system output. The probability of these upset cases depends on several factors.

(i) P_{upset} is the probability of a soft error in the full-precision module, altering its output in some way. This is a function of the environment upset rate and the size of the unmitigated design.

(ii) a is the detection factor, the fraction of upsets which trigger the *reduced-precision mode* in a particular RPR implementation. This factor depends on the *detection capability* of the specific RPR implementation: the type and magnitude of upsets that can be detected.

(iii) P_{fp} is the probability of a false positive detection event, which occurs when RPR erroneously chooses the reduced-precision output over the full-precision output even when the full-precision module was correct. The frequency of occurrence depends on the RPR implementation and the properties of the signals being processed.

Table 1 lists the probabilities of the four upset cases and shows the limit of the noise signal, E_{RPR}, in each case.

3.5. Average RPR Noise Limit. In order to summarize the effect of changing RPR parameters on the performance of the system, we define an *average noise limit* for RPR, $E_{\text{RPR-avg}}$. The average RPR noise limit is based on the probabilities and noise limits of Table 1:

$$\begin{aligned} E_{\text{RPR-avg}} &= \Pr(\text{DU}) \cdot \epsilon_{\max} + \Pr(\text{UU}) \cdot \text{T}_h + \Pr(\text{FD}) \cdot \epsilon_{\max} \\ &= P_{\text{upset}} \cdot a \cdot \epsilon_{\max} + P_{\text{upset}} \cdot (1-a) \cdot T_h \\ &\quad + \left(1 - P_{\text{upset}}\right) \cdot P_{\text{fp}} \cdot \epsilon_{\max}. \end{aligned} \tag{5}$$

This takes into account the probability of occurrence of each upset case and gives an average value of the noise limit E_{RPR} over time.

4. Example System and Experimental Configuration

The discussion offered in this paper is kept as general as possible. When appropriate, however, an example system is used to illustrate the concepts presented and to provide practical demonstrations. This example system is a digital communications circuit. Specifically, a simple demodulator is implemented on an FPGA and is assumed to be operating in a radiation environment.

Table 1: Summary of the possible upset cases for a general RPR module.

Upset case	Probability	Noise signal added	Absolute noise limit (E_{RPR})
DU	$P_{\text{upset}} \cdot a$	ϵ_e	$\epsilon_{\max}$
UU	$P_{\text{upset}} \cdot (1-a)$	ϵ_u	T_h
FD	$(1-P_{\text{upset}}) \cdot P_{\text{fp}}$	ϵ_e	$\epsilon_{\max}$
NU	$(1-P_{\text{upset}}) \cdot (1-P_{\text{fp}})$	0	0

The architecture of the demodulator and the effects of soft errors on the system are used to illustrate the points made in this paper. Fault injection experiments simulated the effects of radiation and provided the data gathered. This section briefly describes the example system, the method of testing the system using fault injection, and the classification of upsets seen in the experiments.

4.1. Example System. Figure 2 shows the block diagram of a simple binary pulse amplitude modulation (PAM) communications system with a Gaussian noise channel. The binary PAM system is the basis for many complex systems including other PAM systems and phase-shift keying (PSK) systems. The demodulator portion of the system is the focus of the analysis and fault injection experiments reported on here.

The matched filter was implemented as a 25-tap FIR filter with symmetric coefficients, which allows the filter to be implemented with 13 multipliers. The filter used a square-root-raised-cosine (SRRC) pulse shape with excess bandwidth $\alpha = 0.5$ using $L_p = 3$ [17]. The matched filter operated at $N = 4$ samples/bit and used 16-bit coefficients. The inputs and filter registers had the same bit widths as the coefficients.

The other two blocks of the demodulator circuit are much simpler than the matched filter. The downsample block passes on every fourth sample and throws away the rest. The decision block is a simple threshold detector, checking the sign of each sample to determine if a 1 or 0 was most likely to have been originally sent based on the received data.

The matched filter makes up the bulk of the demodulator portion of the system in terms of FPGA resources. (The downsample block is simply an enabled register, and the decision block reads and inverts the MSB of the downsample block output as a comparison against zero in two's complement arithmetic). To simplify the analysis of the fault injection results, the filter was the only block implemented on the test FPGA.

4.2. Fault Injection Experiments. Fault injection experiments were used to test the effect of SEUs on the matched filter and on the functionality of the demodulator system. The fault injection experiments were conducted as follows.

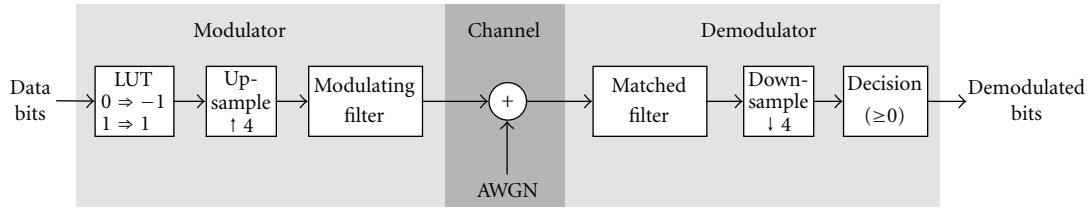

Figure 2: Model of a binary pulse amplitude modulation (PAM) communications systems with an additive white Gaussian noise (AWGN) channel.

(1) The FIR filter design was targeted to a Xilinx Virtex 4 SX-55 FPGA (the DUT FPGA).

(2) The *sensitive* bits of the filter (those FPGA configuration bits which affect a particular design) were identified according to the method described in [18].

(3) One of the bits in the set defined in Step (2) was inverted in the original, clean configuration bit file, and the FPGA was configured using this corrupt file.

(4) For this configuration upset, a bit error rate curve was generated by processing the modulated signal from the FuncMon with the system defined by the corrupted configuration bit file.

(5) For the noncatastrophic SEUs, the bit error rate curve produced by the previous step was compared to the curve for the system in the absence of upsets. The performance loss (in terms of SNR) is estimated by taking the difference of the SNR value of each curve at a bit error rate of 10^{-5}.

Steps (3) through (5) were repeated for each of the sensitive configuration bits, as defined in Step (2). This process simulated the occurrence of all relevant SEUs, each being present one at a time as expected in an FPGA system with a proper scrubbing system.

4.3. SEU Classes. The fault injection experiments resulted in different types of errors in the system, depending on the particular configuration bit upset. We divided the upsets into what we consider to be four types of effects [19]. We label these SEU categories "Class 1 SEU" through "Class 4 SEU."

(1) A Class 1 SEU causes virtually no perturbation in the bit error rate performance of the matched filter detector. The measured loss is less than 0.2 dB, allowing for measurement error of the SNR loss value.

(2) A Class 2 SEU degrades the bit error rate performance in the same way an additional source of additive noise degrades performance.

(3) A Class 3 SEU produces an unusably high bit error rate floor. These SEUs are considered *catastrophic*.

(4) A Class 4 SEU produces a bit error rate of 1/2. These SEUs are also *catastrophic*.

Tables 3 and 6 report the results of the fault injection experiments, tallying the number of SEUs in each of these four classes.

5. Threshold Selection

As described in Section 3, T_h is the error detection threshold of RPR. T_h is an important parameter which controls the magnitude of errors that are detected by RPR. This value controls the noise limits of the RPR output.

In previous work, T_h was set to the maximum estimation error, $\epsilon_{\max}$, as suggested by Shim et al. [11] and as used in [19]. Shim's T_h value is the optimal value in the general case, where the probability distribution of the estimation error signal is unknown. If the designer of a particular system has additional information about this ϵ_e signal, however, a lower threshold value may offer better RPR performance.

This section describes the factors involved in setting the value of T_h and suggests a method for obtaining higher performance with a value of $T_h < \epsilon_{\max}$ for a fixed B_r value. This novel method is made possible by limiting the scope of the RPR implementation to a particular system and will not offer higher performance for *all* systems. Fault injection experiments then demonstrate the potential benefit of these new T_h values.

5.1. Reduction of T_h. The value for T_h affects both the distribution of UU and DU events as well as the noise limits for each of these event types. This shift is represented by the change in the value a in (5) (Table 5 reports on some measured values of the a factor for changing T_h values.) Increasing T_h causes more UU events and fewer DU events, decreasing a. Decreasing T_h has the opposite effect. Decreasing T_h also affects the noise limit in the UU upset case, as seen in the second term of (5). This makes it difficult to determine the overall effect of altering T_h on $E_{\text{RPR-avg}}$.

A low value of T_h (lower than $\epsilon_{\max}$) is desirable because it lowers the noise limit in the UU case. However, there are two possible disadvantages to a lower T_h value.

(1) There are possible false-positive error detection events, as discussed earlier. This introduces noise equal to ϵ_e even when no upsets exist in the system.

(2) Upsets that cause errors with magnitude above T_h but below $\epsilon_{\max}$ are replaced with the estimation error which has a bound at $\epsilon_{\max}$. The resulting error, then, could be larger than the error caused by the upset itself in some cases.

In each of these cases, the RPR system introduces a higher-magnitude noise than would otherwise be present (in the

unmitigated module). Each of these cases will now be described in detail.

5.1.1. False Positive Error Events. In previous work, T_h was set to the maximum estimation error, $\epsilon_{\max}$ to ensure that the false detection upset case did not occur [11, 19]. If the probability of a false detection event, Pr(FD), is sufficiently small, however, it may be desirable to lower T_h to allow some false positive events. Knowledge of the input signal characteristics or the operating environment could allow one to predict Pr(FD) for lower T_h values. Similarly, knowledge of the statistical properties of the ϵ_e signal directly can provide enough information to be able to lower T_h to obtain a better $E_{\text{RPR-avg}}$.

In some cases, with knowledge of the input signal and the properties of a specific module, it is possible to choose $T_h < \epsilon_{\max}$ to avoid false positive detection events a large portion of the time. In this case, $\Pr(\text{FD}) \ll 1$, but may be nonzero. This alters the final term in (5), which is zero when using $T_h = \epsilon_{\max}$ since $\Pr(\text{FD}) = 0$. However, the first and second terms are also altered since the value a is dependent on T_h and T_h itself is the noise limit in the UU case. Without knowing the value of a as a function of T_h, it is difficult to predict the effect on $E_{\text{RPR-avg}}$. This function is dependent on the specific module being protected and the upset environment and is difficult to generalize.

A more direct method is to examine the distribution of the estimation error signal, ϵ_e. Shim and Shanbhag showed that, for a uniformly distributed ϵ_e signal, the optimal value for T_h is $\epsilon_{\max}$ [12]. This is reasonable because all values of ϵ_e between 0 and $\epsilon_{\max}$ are equally probable, including those above any value T_h less than $\epsilon_{\max}$. Thus P_{fp} increases sharply as T_h is lowered below $\epsilon_{\max}$. This, in turn, increases the frequency of the FD upset event which decreases the overall performance of RPR.

If, on the other hand, the distribution of the ϵ_e signal is such that higher values of ϵ_e are less probable than lower values, the increase in P_{fp} may not be enough to severely affect the performance of the system. For example, if the distribution of ϵ_e is Gaussian (the actual ϵ_e signal cannot be a true Gaussian, of course. The ϵ_e signal has an actual cutoff at $\epsilon_{\max}$ while a true Gaussian distribution has infinite support.) the false error probability can be predicted based on the relation of T_h to the standard deviation (σ) of the distribution. Table 2 shows the relation of P_{fp} to T_h for this case. A system with $T_h = \sigma$ can expect a false positive every third clock cycle, on average. Values of $T_h = 5\sigma$ and $T_h = 6\sigma$, however, result in false positive error rates of less than 10^{-6}. With rates this low, it can certainly be feasible to lower T_h without fear of significantly increasing the FD upset case probability.

The distribution of ϵ_e is highly dependent on the type of module being protected as well as the signal environment at its input. Consider, for example, the FIR filter module of Section 4.1 and its submodules: registers, adders, and multipliers. A simple register with a uniformly distributed input would have a uniformly distributed ϵ_e signal. In our testing, a constant coefficient multiplier showed varying distributions for ϵ_e based on the coefficient value and the B_r value. For each of these combinations, a different amount of truncation occurred in the coefficient resulting in several error distributions. These included distributions that appeared approximately uniform, Gaussian, or triangular. For a full FIR filter with a modulated input signal, however, the ϵ_e signal appeared Gaussian when the input signal had a signal-to-noise ratio (SNR) less than 30 dB [16]. This property is exploited in Section 5.2 in order to find a valid $T_h < \epsilon_{\max}$.

Table 2: P_{fp} values for a Gaussian-distributed ϵ_e signal.

T_h	P_{fp}
σ	0.317
2σ	0.0455
3σ	2.70×10^{-3}
4σ	6.33×10^{-5}
5σ	5.73×10^{-7}
6σ	1.97×10^{-9}

5.1.2. Midrange Upset Errors. The second problem mentioned with lowering T_h below $\epsilon_{\max}$ is the possible increase in the error level for some upsets. In this case, the noise induced by some upsets will be replaced by the noise of the RP_{out} signal: ϵ_e. This results in the $\epsilon_{\max}$ value being the noise limit a higher percentage of the time while the *reduced threshold value*, T_h^*, is the noise limit a lower percentage of the time. Depending on the noise induced by the SEU, this could result in a higher overall noise level.

For example, consider the probability mass functions (pmf) shown in Figure 3 representing some error signals of a hypothetical RPR system (The pmfs displayed were created to be zero-mean Gaussian distributions for illustration purposes. It is important to note that these error signals do not always have this type of distribution.) Figure 3(a) shows the pmf of the estimation error signal, ϵ_e, of an RPR module along with its noise limit, $\epsilon_{\max}$. Figure 3(b) shows the pmf of the upset error signal, ϵ_u, of the SEU with the largest undetected error signal for a given reduced threshold, T_h^*. Figure 3(c) shows the pmf of another upset error signal for which the maximum value of ϵ_u is $T_h^* < \epsilon_{u\text{-max}} < \epsilon_{\max}$.

In the case of Figure 3(c), the upset causes noise higher than T_h^* and is detected as an error. The RPR system thus enters the reduced-precision mode and the error signal of Figure 3(c) is replaced with that of a reduced-precision module as shown in Figure 3(a). In this case, the error of the system is increased due to the lowered threshold value.

This discussion shows that the effect of lowering T_h below $\epsilon_{\max}$ can have mixed consequences. With additional knowledge about a specific system (including characteristics of the input signal, the noise induced by each upset, and the estimation error of the reduced-precision modules) it would be possible to predetermine the optimal value for T_h. In the end, however, the most general acceptable rule is that T_h should not be lowered below $\epsilon_{\max}$, as stated by Shim. With that in mind, the following section introduces a method for

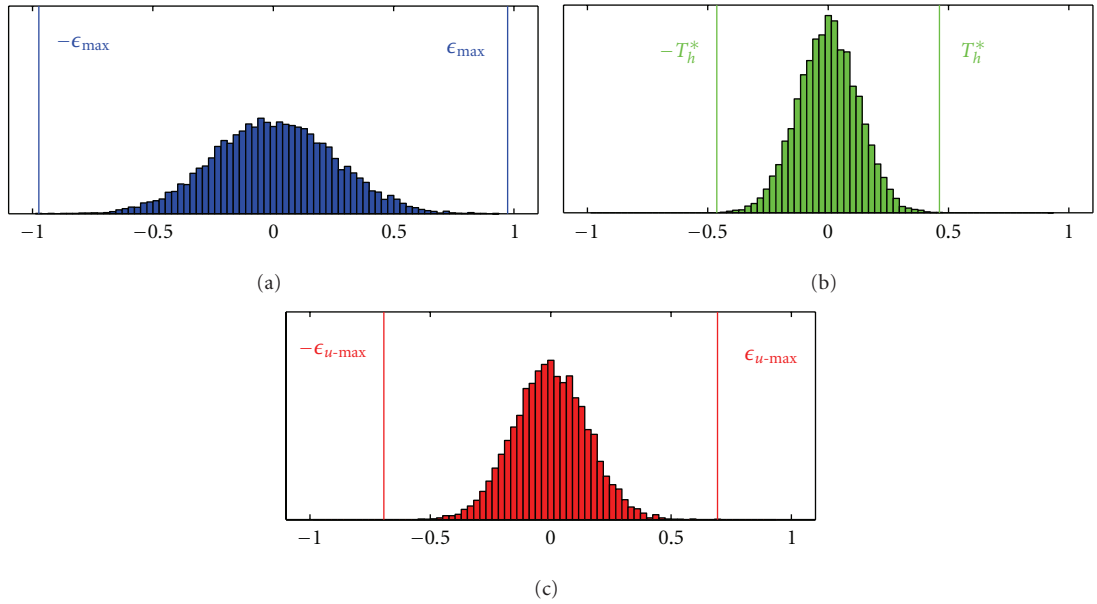

FIGURE 3: (a) The pmf of the estimation error, ϵ_e, of an RPR module, (b) the pmf for the maximum undetected upset error signal, ϵ_u, and the pmf for (c) a midrange upset which crosses the reduced threshold, T_h^*.

finding an acceptable lower value for T_h experimentally for certain systems.

5.2. Experimental Determination of T_h. Although the theoretical value of $T_h = \epsilon_{\max}$ is sure to avoid the negative issues presented in Section 5.1, this value may be higher than necessary for T_h in practice. Rather than using the theoretical value, the maximum value of ϵ_e can be determined experimentally. We label this experimentally determined value $\epsilon^*_{\max}$, which is used to determine the experimental decision threshold labeled T_h^*, where $\epsilon^*_{\max} < \epsilon_{\max}$ and $T_h^* < T_h$.

For the FIR filter circuit, we have experimentally measured the signal ϵ_e for several different RPR bit widths. To do this, we created bit-accurate simulation models of the full-precision and reduced-precision FIR filter circuits using Matlab. We then generated several representative modulated input signals, each with a different SNR level (SNR values of 2, 4, 6, 8, and 10 dB). These models were then used as follows.

(1) Each of the input signals was processed by the FP filter and the output signals recorded.

(2) The same input signals were processed by each RP filter and the output signals recorded.

(3) For each RP filter and each SNR, the estimation error signal, ϵ_e, was calculated.

(4) The absolute maximum value of each ϵ_e signal was recorded as $\epsilon^*_{\max}$.

(5) The mean (μ_e) and standard deviation (σ_e) of each ϵ_e signal were calculated.

For this design and these input characteristics, the signal ϵ_e was roughly Gaussian distributed, though not with a mean of zero as in the examples in Figure 3 (The nonzero mean of these error signals is due to the truncation of the signals associated with the reduced-precision module. The truncation operation introduces a positive error bias to the error signal ϵ_e.) As expected, the $\epsilon^*_{\max}$ value was dependent on the test duration. We also discovered that the SNR of the input signal did not have a significant impact on the statistics of the ϵ_e signal.

Using the Gaussian distribution of ϵ_e and the values in Table 2 as a hint, we calculated the experimental threshold as

$$T_h^* = \mu_e + 6\sigma_e. \tag{6}$$

We confirmed this to be a valid threshold (i.e., $T_h^* > \epsilon^*_{\max}$) for simulation durations up to 10^6 samples. With this value of T_h^*, we expected P_{fp} to be very low in practice, as suggested by Table 2.

Table 4 shows the different threshold values obtained for several different reduced-precision FIR filters. Both the theoretical (T_h) and experimental (T_h^*) threshold values are shown for each filter as well as the mean (μ) and standard deviation (σ) values for the signal ϵ_e. Notice that the experimentally-determined threshold values, in these cases, become increasingly lower than their theoretical counterparts as B_r decreases. This can greatly increase the number of errors detected for a particular bit width and has the potential to make even lower B_r values feasible for a particular system, decreasing the area overhead of RPR.

The T_h values shown in the table are the calculated maximum values of ϵ_e [16]. The next sections will present experimental results for designs using both the T_h and T_h^* values. The results will show that the lowered threshold values can have a significant impact on the performance of RPR, especially for the lower values of B_r tested.

Table 3: Number of SEUs causing each class of effect for an FIR filter protected with TMR and several levels of RPR using experimentally determined thresholds (T_h^*), compared to mathematically determined thresholds (T_h).

Design	Slices used	Class 1 bits	Class 2 bits	Class 3 bits	Class 4 bits	Total catastrophic (% reduction)	Improv. in failure rate
Unmitigated	1,030	59,156	6,472	1,501	943	2,444 (—%)	—
RPR, $B_r = 7$, T_h	1,755	106,751	6,239	11	2	13 (99.47%)	188×
RPR, $B_r = 7$, T_h^*	1,755	106,863	6,191	11	2	13 (99.47%)	188×
RPR, $B_r = 5$, T_h	1,470	84,284	7,819	226	2	228 (90.67%)	10.7×
RPR, $B_r = 5$, T_h^*	1,470	84,583	7,709	42	2	44 (98.20%)	55.5×
RPR, $B_r = 3$, T_h	1,313	73,992	6,875	1,598	666	2,264 (7.36%)	1.08×
RPR, $B_r = 3$, T_h^*	1,313	74,129	8,267	634	36	670 (72.59%)	3.65×

Table 4: Mathematical (T_h) versus experimental (T_h^*) threshold values for RPR FIR filter designs with several different reduced-precision bit widths (B_r). The mean (μ_e) and standard deviation (σ_e) values for the signal ϵ_e are also shown.

B_r	T_h	T_h^*	% Change	μ_e	σ_e
7	0.1597	0.1049	−34.3%	0.05380	0.008500
6	0.3106	0.1844	−40.6%	0.08365	0.01563
5	0.6046	0.3182	−47.4%	0.1431	0.02891
4	1.2212	0.5849	−52.1%	0.2453	0.0562
3	2.3871	0.9222	−61.4%	0.3659	0.09465

5.3. Reduced Threshold Experiments. To demonstrate the effects of using the experimentally determined T_h^* values, fault injection experiments were run on a set of FIR filter designs. The configuration of these experiments was as described in Section 4.2. Three levels of RPR were implemented using $B_r = 3$, 5, and 7.

Table 3 shows the results of these experiments. The results are presented as in a previous paper [19], categorizing the SEUs into four classes, as explained in Section 4.3.

Notice that there was no change in the number of catastrophic upsets for $B_r = 7$, which had the smallest percentage change from T_h to T_h^* shown in Table 4. For the lower B_r values, the difference in threshold value is larger and the effect on performance is greater. The coverage of catastrophic errors increased by 8% for $B_r = 5$ and by 65% for $B_r = 3$.

Table 5 reports on measured values of the RPR detection factor, a, for both threshold values. This value is the fraction of upsets in the full-precision module that were detected by the RPR system and for which the reduced-precision output was used. Note that, as expected, the a factor increases with the lower threshold T_h^* for each B_r value.

Table 5: Detection factor (a) for an FIR filter protected with several levels of RPR using experimentally determined thresholds (T_h^*), compared to mathematically determined thresholds (T_h) at an SNR of 8 dB.

Design	a
RPR, $B_r = 7$, T_h	0.0754
RPR, $B_r = 7$, T_h^*	0.1082
RPR, $B_r = 5$, T_h	0.0519
RPR, $B_r = 5$, T_h^*	0.0859
RPR, $B_r = 3$, T_h	0.0495
RPR, $B_r = 3$, T_h^*	0.0699

6. Bit-Width Selection

The previous section discussed setting T_h for a fixed reduced-precision bit width, B_r. This section presents the considerations necessary when setting B_r. The value of B_r determines the quality of the estimate that the reduced-precision modules produce relative to the full-precision module. This in turn controls the valid range of T_h and the level of noise that is detectable by the system.

In general, a higher B_r has a higher area cost and gives better performance. A higher B_r gives a better estimate of the full-precision output, resulting in a lower and smaller range for T_h. The effect on performance can be seen in (5); since both $\epsilon_{\max}$ and T_h decrease with an increase in B_r, the average noise limit of RPR decreases as well.

This section emphasizes that the selection of B_r has a large impact on the performance and cost of RPR. It describes this impact and presents how to calculate the valid range of B_r available for a particular module. It also demonstrates the trade-offs between the cost and performance factors with fault injection experiments.

6.1. Bit-Width Effects. The primary effect of setting B_r is to set the accuracy of the estimate of the full-precision module and thus the estimation error signal, ϵ_e. This affects not only the noise of the system in *reduced-precision mode*, but also the level of SEU-induced noise that is detectable.

6.1.1. Effect on Performance. The B_r value directly sets the noise level of the RPR system while it is in *reduced-precision mode*. RPR operates in this mode when an error is detected in the full-precision module and the reduced-precision output is used. Thus the noise level in this mode depends solely on

Table 6: Number of SEUs causing each class of effect for an FIR filter protected with TMR and several levels of RPR using experimentally determined thresholds (T_h^*), compared to the unmitigated filter.

Design	Slices used	Slices overhead	Class 1 bits	Class 2 bits	Class 3 bits	Class 4 bits	Total utilized bits	Total catastrophic (% reduction)	Improv. in failure rate
Unmitigated	1,030	—	59,156	6,472	1,501	943	68,072	2,444 (—%)	—
TMR	3,171	208%	218,304	0	0	2	218,306	2 (99.9%)	1222×
RPR, $B_r = 7$	1,755	70.4%	106,863	6,191	11	2	113,067	13 (99.5%)	188×
RPR, $B_r = 6$	1,602	55.5%	95,980	7,731	9	2	103,722	11 (99.6%)	222×
RPR, $B_r = 5$	1,470	42.7%	84,583	7,709	42	2	92,336	44 (98.2%)	55.5×
RPR, $B_r = 4$	1,394	35.3%	79,334	8,252	254	2	87,842	256 (89.5%)	9.55×
RPR, $B_r = 3$	1,313	27.5%	74,129	8,267	634	36	83,066	670 (72.6%)	3.65×

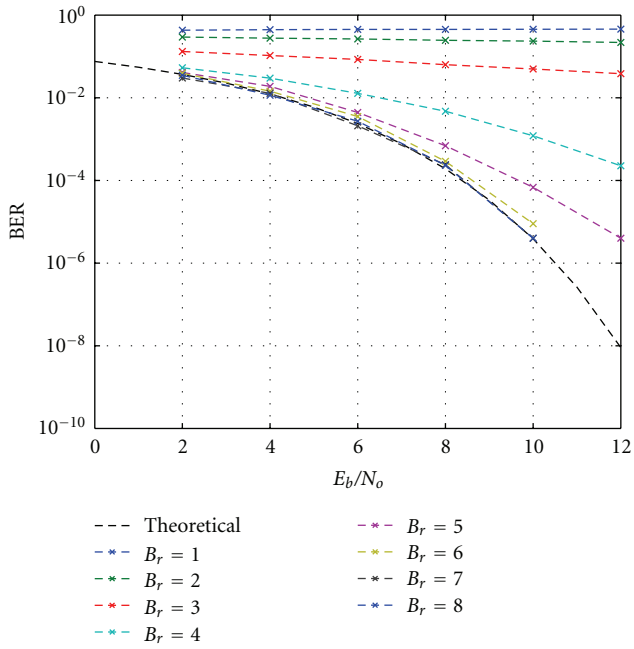

Figure 4: Bit error rate curves for several FIR filters with different bit-widths.

the performance of the reduced-precision module, which is dependent upon its bit width.

For example, Figure 4 shows several BER curves for the binary PAM system described in Section 4.1, each for an FIR filter with a different input bit width. If one of the application requirements specifies that the BER in reduced-precision mode should be at most 10^{-4} at an SNR of 10 dB, the input bit width of the RP modules must be $B_r \geq 5$.

The B_r value also controls the level of SEU-induced noise that is detectable. A smaller B_r value means that the reduced-precision module produces a poorer estimate of the full-precision output, resulting in a larger possible difference between the two outputs. Thus a higher threshold, T_h, is needed for a smaller B_r.

6.1.2. Effect on Error Detection Threshold. Lowering the B_r value decreases the performance of an RPR system, resulting in a cutoff of its usefulness as B_r approaches zero. As B_r is lowered, T_h must become larger. Obviously, there are few interesting circuits that would be estimated well by a reduced-precision module with $B_r = 0$ (a 1-bit signed number). Depending on the application, the value for T_h could be too large to be usable even at B_r values significantly higher than 0.

Using the binary PAM system as an example, the output of the full-precision FIR filter has a bit width of $B + 1 = 16$ with a range of $[-2, 2)$. From Table 4, the theoretical value of T_h for $B_r = 3$ is 2.3871. This is over 50% of the total range of the output signal of the filter. In fact, the output range of the filter is typically smaller than this.

As an example of a system with a valid threshold, Figure 5 gives a representation of the signals used by the RPR decision block to determine if there is an error in the system. This figure was generated from the outputs of an RPR FIR filter with $B_r = 6$ and $T_h = 0.3106$ and no errors present. By adding and subtracting T_h to and from the RP_{out} signal, the upper and lower bounds for the FP_{out} signal can be visualized. Note that in this system, the noise limits are fairly close to the full-precision output. An error in the full-precision module which caused the output to exit these bounds would be flagged as an error and the reduced-precision output would be used instead.

By adding and subtracting $T_h = 0.3106$ to and from the RP_{out} signal, the upper and lower bounds for the FP_{out} signal can be visualized. In contrast, Figure 6 shows the signals for the FIR filter with $B_r = 3$ and $T_h = 2.3871$. The figure illustrates the system with a catastrophic error in the full-precision module: FP_{out} is frozen at 0. With this value of T_h, the erroneous FP_{out} signal is always completely within the displayed bounds. Thus the RPR decision block determines that no error is present in the full-precision module and uses the frozen output as RPR_{out}.

This T_h value is too large to handle this type of error. This type of error is fairly common for this FPGA design when the clock or reset line is upset. This explains the poor performance of RPR with $B_r = 3$ and $T_h = 2.3871$ in terms of preventing catastrophic errors as reported in Table 3. For

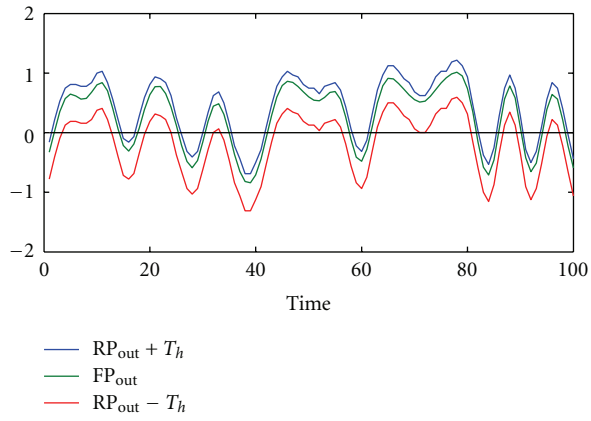

FIGURE 5: RPR filter decision signals for RPR with $B_r = 6$ and $T_h = 0.3106$. No errors are present in the system. The upper and lower comparison bound signals are calculated by adding and subtracting T_h to and from RP$_{out}$.

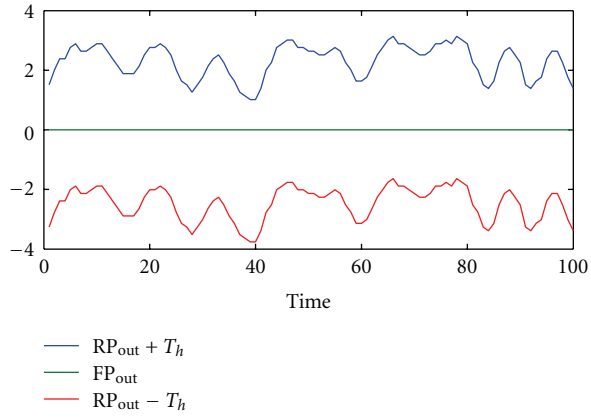

FIGURE 6: RPR filter decision signals for RPR with $B_r = 3$ and $T_h = 2.3871$. The FP$_{out}$ signal is frozen at zero. The upper and lower comparison bound signals are calculated by adding and subtracting T_h to and from RP$_{out}$.

this design, then, a larger B_r value must be used to give adequate performance. With a larger B_r and a lower T_h value, the frozen full-precision output would be more likely to be outside the noise limits. Using the theoretical T_h values, a bit-width of $B_r = 6$ or $B_r = 7$ would be more appropriate for a signal with this output range.

6.2. General Bit-Width Selection. Selecting the best value of B_r is highly dependent on the application in question. This section presents a general overview of selecting possible B_r values for an RPR module.

6.2.1. Upper Bound. The upper bound of B_r depends on several factors. The most obvious of these is $B_r < B$ (the full-precision bit width) since $B_r = B$ is essentially TMR, which gives full protection against single upsets. Even values close to B are undesirable due to the increased overhead of the large RPR decision blocks compared to minimal TMR voters.

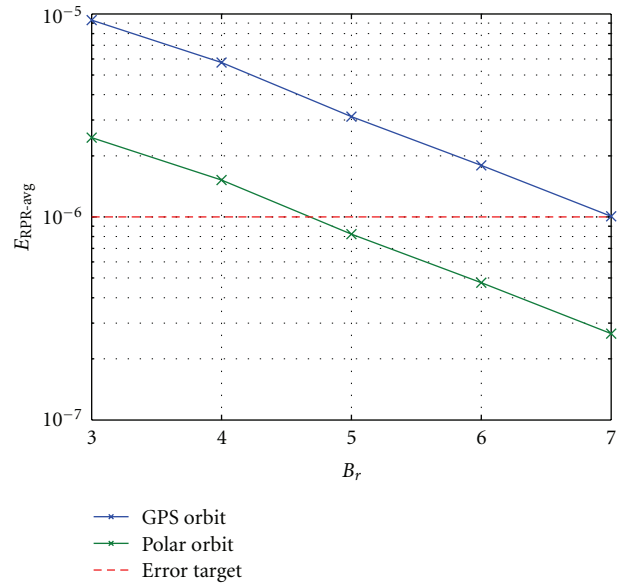

FIGURE 7: $E_{RPR\text{-}avg}$ of the FIR filter design for several bit widths and using two failure rates.

Another simple upper bound is an area or power limit imposed by application constraints. Besides the area and power costs of higher B_r values, there is no general downside to increased precision in the reduced-precision modules. This can only increase the performance of the RPR system.

6.2.2. Lower Bound. The lower bound of B_r is determined by the point at which the detection capabilities of RPR degrade to unusable levels. Section 6.1 described an example where a low B_r value caused the T_h value to increase such that critical errors went undetected. Similar methods can be used for other systems.

In a more general sense, the T_h value is the general noise limit on the RPR system, as seen in (5). The designer of the RPR module can thus define an acceptable noise limit at the output of the RPR decision block and increase B_r until the calculated or measured value of T_h falls below this bound.

6.2.3. Optimization. These bounds, of course, are only a starting point for selecting B_r for a particular module. At this point, the designer must find the optimal trade-off between the cost of implementation and the performance of the system. If the upset rate of the target environment is very low, $E_{RPR\text{-}avg}$ will be small even with a low B_r value. If the upset rate is higher, it may be more important to use a high B_r value to keep the noise low in the DU upset case.

For example, Figure 7 plots the value of $E_{RPR\text{-}avg}$ of the FIR filter design for several bit widths in two different upset environments: GPS orbit and Polar orbit (the upset rates for these orbits and this filter design are available in [16].) If the target $E_{RPR\text{-}avg}$ for this system is 10^{-6}, the system in the Polar orbit requires a B_r of 5. With the higher upset rate of the GPS orbit, however, the system requires a B_r of at least 7 to meet the noise limit target.

In this case, using $E_{\text{RPR-avg}}$ as the measure of performance of the RPR system, the upsets are not frequent enough in the Polar orbit to warrant a high cost of RPR. In the GPS orbit, however, the RPR system is predicted to enter *reduced-precision mode* much more often, increasing $E_{\text{RPR-avg}}$ significantly.

The effects of these trade-offs are highly dependent on the application in question and cannot be generalized. What is important is that RPR can give many options for increasing the performance of a system in the presence of SEUs. The next section presents results from fault injection experiments that demonstrate these options, which trade-off circuit area for performance.

6.3. Bit-Width Experiments. In order to demonstrate the effects of varying the reduced-precision bit width (B_r) for RPR, the fault injection experiments of Section 5.3 were expanded. This section reports on the performance of the simple communications system of Section 4.1 for $B_r = 3$ to 7. The designs tested used the experimentally determined thresholds T_h^* in Table 4. The results emphasize the flexibility of RPR by demonstrating the wide range of cost and performance trade-off points that RPR offers this system.

Table 6 shows the SEU classification results from the fault injection experiments. As expected, increasing the bit width of the reduced-precision filters improved the handling of catastrophic SEUs. The cost of implementation increased with B_r as well.

The SEUs may also be quantified by the SNR loss they cause at the output of the filter. These results are summarized in Figure 8. These data define a cumulative distribution of the SNR loss for each of the 6 designs at a bit error rate of 10^{-5}. (Note that Class 3 and Class 4 SEUs have infinite SNR loss and are included in the percentages shown.) As an example, consider the unmitigated filter design. Approximately 9% of all SEUs within the filter circuit lead to an SNR loss in excess of 1 dB. In other words, 91% of all the SEUs affecting the filter give an SNR loss less than 1 dB.

Figure 8 plots the SNR loss values for the various versions of this filter. Notice that the increase in B_r does more than increase the design's resistance to catastrophic SEUs. As the size of the reduced-precision filters increases, the number of higher-noise SEUs decreases as well. As expected, the more costly the RPR system, the lower the overall noise and the higher the performance.

TMR was much more effective at protecting the receiver system against SEUs than RPR in our experiments. However, in the case of the RPR implementation with $B_r = 6$, the overhead cost of implementing RPR was about one quarter that of TMR. This version of RPR reduced the number of catastrophic bits by over 99% and significantly reduced the number of high-noise SEUs. Although the RPR implementation with $B_r = 7$ did not offer any improvement in protection against catastrophic SEUs over the $B_r = 6$ design, Figure 8 reflects the improvements in SNR loss offered by the extra hardware required. Even the implementation with $B_r = 3$ offers a significant improvement: at a cost of only 28% more hardware, the number of catastrophic bits decreased by over 70%.

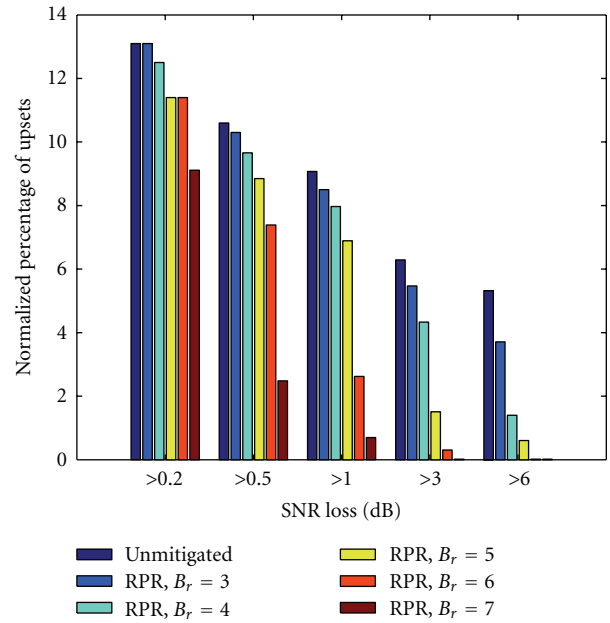

Figure 8: Normalized percentage of SEUs causing certain SNR losses at a BER of 10^{-5} for an FIR filter protected with several levels of RPR compared to the unmitigated design.

These results emphasize that RPR offers flexibility in its implementation options. It is fairly straightforward to increase the performance of an RPR system in the presence of SEUs by increasing the amount of redundancy in the reduced-precision modules. The range of options RPR offers a particular application depends on the system to be protected and the application requirements. It is clear, however, that RPR can offer intriguing trade-offs between cost and performance.

7. Conclusion

This paper has confirmed that reduced-precision redundancy has great potential to reduce the cost of soft error mitigation in FPGA-based circuits. Experiments shown here demonstrate improvements in failure rate over an unmitigated system by as much as a 200 times at less than half the area overhead cost of TMR.

As a further contribution, this paper provides an in-depth analysis of the parameters involved in using RPR: the error detection threshold, T_h and the reduced-precision bit width (B_r). The discussion and examples provided emphasize the effects of these parameters on the size and performance of the resulting system. Detailed fault injection experiments and reports on the area cost of RPR give greater insight into the actual results of implementing RPR in an FPGA system. In addition, an experimental method for improving the performance of RPR under certain conditions by optimizing the T_h parameter for a particular system was presented. This was shown to result in an improvement of

up to 5 times at no additional hardware cost over the original RPR implementation.

Although the examples given in this paper are FPGA-based systems with the intent of masking the effects of SEUs, the RPR technique can, of course, be expanded further. Fault-masking techniques such as TMR as well as error-reducing techniques such as RPR can also protect against the lesser transient and soft data errors. In addition, RPR can be applied outside of SRAM-based FPGA systems, just as TMR has been in many instances. The insights into the implementation of RPR presented here can also be utilized in the protection of the more robust ASIC and in other FPGA technologies. Future work could include similar detailed experimental analysis on ASIC-based circuits as well as other types of circuit structures aside from the digital filter example presented here.

Acknowledgment

This work was supported by the I/UCRC Program of the National Science Foundation under Grant no. 0801876.

References

[1] M. Cummings and S. Haruyama, "FPGA in the software radio," *IEEE Communications Magazine*, vol. 37, no. 2, pp. 108–112, 1999.

[2] M. Caffrey, "A space-based reconfigurable radio," in *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA '02)*, M. Caffrey, T. P. Plaks, and P. M. Athanas, Eds., pp. 49–53, CSREA Press, June 2002.

[3] Z. K. Baker, M. E. Dunham, K. Morgan et al., "Space-based FPGA radio receiver design, debug, and development of a radiation-tolerant computing system," *International Journal of Reconfigurable Computing*, vol. 2010, Article ID 546217, 12 pages, 2010.

[4] Y. Zhang, L. Chang, G. Yang, and H. Li, "Reconfigurable-system-on-chip implementation of data processing units for space applications," *Transactions of Tianjin University*, vol. 16, no. 4, pp. 270–274, 2010.

[5] A. K. Brown and N. Thompson, "Dynamically reconfigurable software defined radio for GNSS applications," in *Proceedings of the Software Defined Radio Forum (SDR '10)*, pp. 538–542, 2010.

[6] T. Karnik and P. Hazucha, "Characterization of soft errors caused by single event upsets in CMOS processes," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 2, pp. 128–143, 2004.

[7] P. E. Dodd and L. W. Massengill, "Basic mechanisms and modeling of single-event upset in digital microelectronics," *IEEE Transactions on Nuclear Science*, vol. 50, no. 3, pp. 583–602, 2003.

[8] C. Carmichael, "Triple module redundancy design techniques for Virtex FPGAs," Tech. Rep. XAPP197 (v1.0), Xilinx Corporation, 2001.

[9] N. Rollins, M. Wirthlin, M. Caffrey, and P. Graham, "Evaluating TMR techniques in the presence of single event upsets," in *Proceedings of the 6th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD '03)*, p. 63, NASA Office of Logic Design, AIAA, Washington, DC, USA, Sept. 2003.

[10] B. Shim and N. R. Shanbhag, "Reduced precision redundancy for low-power digital filtering," in *Proceedings of the Conference Record of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers,*, pp. 148–152, November 2001.

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

[12] B. Shim and N. R. Shanbhag, "Energy-efficient soft error-tolerant digital signal processing," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 4, Article ID 1637464, pp. 336–348, 2006.

[13] P. Reviriego, J. A. Maestro, and S. F. Liu, "Efficient soft error-tolerant adaptive equalizers," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 57, no. 8, Article ID 5418885, pp. 2032–2040, 2010.

[14] J. Snodgrass, *Low-power fault tolerance for spacecraft FPGA-based numerical computing*, Ph.D. thesis, Naval Postgraduate School, Monterey, Calif, USA, 2006.

[15] M. A. Sullivan, *Reduced precision redundancy applied to arithmetic operations in field programmable gate arrays for satellite control and sensor systems*, M.S. thesis, Naval Postgraduate School, Monterey, Calif, USA, 2008.

[16] B. H. Pratt, *Analysis and mitigation of SEU-induced noise in FPGAbased DSP systems*, Ph.D. thesis, Brigham Young University, Provo, Utah, USA, 2011.

[17] M. Rice, *Digital Communications: A Discrete-Time Approach*, Pearson Prentice Hall, NJ, USA, 1st edition, 2009.

[18] E. Johnson, M. Caffrey, P. Graham, N. Rollins, and M. Wirthlin, "Accelerator validation of an FPGA SEU simulator," *IEEE Transactions on Nuclear Science*, vol. 50, no. 6, pp. 2147–2157, 2003.

[19] B. Pratt, M. Fuller, M. Rice, and M. Wirthlin, "Reliable communications using FPGAs in high-radiation environments—Part I: characterization," in *Proceedings of the IEEE International Conference on Communications (ICC '10)*, pp. 1–5, Cape Town, South Africa, May 2010.