Reliable Communications Using FPGAs in High-Radiation Environments – Part I: Characterization

Brian Pratt, Megan Fuller, Michael Rice, Michael Wirthlin
NSF Center for High-Performance Reconfigurable Computing (CHREC)
Department of Electrical & Computer Engineering
Brigham Young University, Provo, Utah, USA

***Abstract*—Reconfigurable radios implemented on FPGAs operating in high-radiation environments are subject to single-event-upsets (SEUs). The traditional mitigation method of applying triple modular redundancy (TMR) to the entire design does not have to be used in this application. This is because the majority of the SEUs impact the overall performance (measured by bit error rate) in the same way additive noise does. The results of this paper show which sections must be protected from SEUs and provide a guide for the bit error rate performance versus FPGA area tradeoff as a function of SEU mitigation.**

I. Introduction

Field-programmable gate arrays (FPGAs) are an attractive target platform for reconfigurable radios [1]. Their ability to combine flexibility with good performance makes FPGAs popular for software-defined radios [2]. FPGAs have been used to implement communication-specific processors for well over a decade. When operating in high-radiation environments, FPGAs are susceptible to harmful effects of the high-energy particles that populate these environments. FPGAs are often considered for space applications where they may be subjected to these high-radiation environments.

SRAM-based (synchronous random access memory) FPGAs are made up of a large array of memory cells. These memory cells hold both user data as well as the configuration of the hardware the FPGA implements. Charged particles affect these cells by occasionally inverting the contents of a particular cell. Such an event is called a "single-event upset" (SEU) [3]. The rate at which SEUs occur is determined by the flux of the high-energy particles. A corrupted memory cell may alter either the user data or the FPGA configuration [4]. In most communication designs, the vast majority of memory cells are dedicated to defining the FPGA configuration. Consequently, a corrupted memory cell alters the hardware function performed by the FPGA. That is, the radiation *alters the circuit more than the data*.

To prevent SEUs from accumulating, a technique known as *scrubbing* is used. With this technique, a copy of the bits that define the FPGA configuration are stored in an external protected memory. This copy is used to periodically refresh the FPGA configuration memory [5]. When the refresh rate is large relative to the mean upset rate, the most likely occurrence is at most one SEU between each refresh. The primary mechanism for mitigating SEUs in commercial off-the-shelf FPGAs in this scenario is triple modular redundancy (TMR) combined with configuration scrubbing. At its core, TMR replicates each hardware module three times and uses a voting scheme to determine the correct output [6]. TMR works well when only a single upset is present in the system at one time, which is the norm when combined with scrubbing.

To a digital circuit designer, a communication system appears as a large digital system composed of memories for a sequential state machine and logic gates for performing the accompanying combinational logic. The figure of merit for a digital circuit is bit-level accuracy. From this point of view, every element in the circuit must be protected using a technique such as TMR, which results in a system in excess of three times the size of the original circuit. In contrast, a communications engineer tends to view a digital communication system as a signal processor designed to tolerate some degree of noise. The figure of merit for a digital communications system is the bit error rate. Motivated by these differences, the question is this: *It may not be necessary to apply mitigation to the entire system to achieve acceptable performance. If a judicious application of mitigation can be applied, where should it be applied?*

As a starting point to find an answer to this question, we examined the performance of a binary PAM (pulse amplitude modulation) system in the presence of uncompensated SEUs and make the following observations:

1) Most of the SEUs corrupt the system in a way that behaves like additive noise (The exact proportions are summarized below). The other SEUs cause severe degradation and are called catastrophic SEUs.
2) The function and location of the catastrophic SEUs is predictable.
3) The variance of the additive noise and the percentage of catastrophic SEUs depends on the matched filter coefficients and the number of bits used to quantize each

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

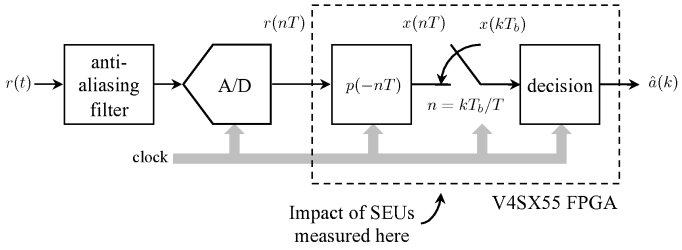

Fig. 1. A high-level block diagram of the system.

filter coefficient.

4) The variance of the additive noise and the percentage of catastrophic SEUs is not strongly dependent on the filter architecture.

Using these observations, we conclude that SEUs in a communications system do not have to be fully protected as in an arbitrary digital circuit. This paper will also describe the location of the catastrophic SEUs, thus providing a guide for engineers wishing to protect similar systems at a lower cost than TMR.

II. System Model and Experimental Configuration

The PAM system is summarized in Figure 1. A sampled-data PAM detector is implemented in the FPGA. The anti-aliasing filter and digitizer (ADC) are outside the FPGA and are assumed to operate normally in the high-radiation environment. For the experiments presented here, the downsample and decision modules were implemented outside the FPGA to simplify the analysis.

A. System Model

The transmitted signal is

$$s(t) = \sum_k a(k)p(t - kT_b) \tag{1}$$

where $a(k) \in \{-1, +1\}$ is the k-th symbol, T_b is the bit time, and $p(t)$ is a unit-energy pulse shape with support $-L_pT_b \leq t \leq L_pT_b$. We assume the pulse shape satisfies the Nyquist No-ISI condition [7] (i.e., it is a "square-root Nyquist pulse"). In the additive white Gaussian noise environment, the received signal is

$$r(t) = \sum_k a(k)p(t - kT_b - \tau) + w(t) \tag{2}$$

where τ is the propagation delay and $w(t)$ is the additive noise modeled as a zero-mean white Gaussian random process. The received signal is sampled by an analog-to-digital converter with clock period T s/sample. We assume that the sample rate $1/T$ and the bit rate $1/T_b$ are commensurate and that $T_b/T = N$ samples/bit. The n-th sample of the received signal is

$$r(nT) = \sum_k a(k)p(nT - kT_b - \tau) + w(nT) \tag{3}$$

$$= \sum_k a(k)p((n - kN)T - \tau) + w(nT) \tag{4}$$

Assuming an ideal anti-aliasing filter, the sequence $w(nT)$ are uncorrelated Gaussian random variables with zero mean and common variance $\sigma_2 = N_0/2$. The received samples are processed by a discrete-time filter whose impulse response is matched to the pulse shape to produce the sequence $x(nT)$. The matched filter output is downsampled by N and the resulting sample $x(kT_b + \tau)$ is used for detection. Because the pulse shape satisfied the Nyquist No-ISI condition and assuming perfect timing synchronization, decision variable $x(kT_b + \tau)$ may be expressed as

$$x(kT_b + \tau) = a(k) + v(kT_b) \tag{5}$$

where the $v(kT_b)$ are a sequence of uncorrelated zero-mean Gaussian random variables with common variance $\sigma^2 = N_0/2$.

B. Experimental Configuration

The impact of SEUs is to alter the FPGA circuit. A fault injection experiment was used to examine the impact of SEUs on system performance. In these experiments, the pulse shape was the square-root raised-cosine (SRRC) pulse shape with excess bandwidth α using $L_p = 6$ [7]. In each case, the matched filter operated at $N = 4$ samples/bit. Filter implementations with 16-bit filter coefficients and 8-bit filter coefficients were examined. Two filter designs were considered.

- A direct form 1 FIR (Finite Impulse Response) filter, as shown in Figure 2 (a), was constructed directly from FPGA slices.
- An alternative approach, based on the built-in DSP blocks (called "dsp48" blocks), was used to design a transposed direct form 1 FIR filter, as illustrated in Figure 2 (b).

Six combinations of these design parameters were investigated:

- "16b logic $\alpha = 1.0$" – direct form 1 filter using 16-bit filter coefficients and a roll-off factor of 1.0.
- "16b logic $\alpha = 0.25$" – direct form 1 filter using 16-bit filter coefficients and a roll-off factor of 0.25.
- "8b logic $\alpha = 1.0$" – direct form 1 filter using 8-bit filter coefficients and a roll-off factor of 1.0.
- "8b logic $\alpha = 0.25$" – direct form 1 filter using 8-bit filter coefficients and a roll-off factor of 0.25.
- "16b dsp48 $\alpha = 1.0$" – transposed direct form 1 filter using 8-bit filter coefficients and a roll-off factor of 1.0.
- "16b dsp48 $\alpha = 0.25$" – transposed direct form 1 filter using 8-bit filter coefficients and a roll-off factor of 0.25.

The fault injection experiments were conducted as follows:

1) The design shown inside the dashed box of Figure 1 was targeted to a V4SX55 FPGA. The design is defined by a file called a "configuration bit file."
2) The bits in the configuration bit file define the contents of all memory cells in the entire FPGA. The bits defining the memory cell contents corresponding to the matched filter of Figure 1 were identified [4]. [1]

[1]The downsample and decision blocks were ignored in these experiments. The filter makes up the bulk of the design in terms of configuration bits.

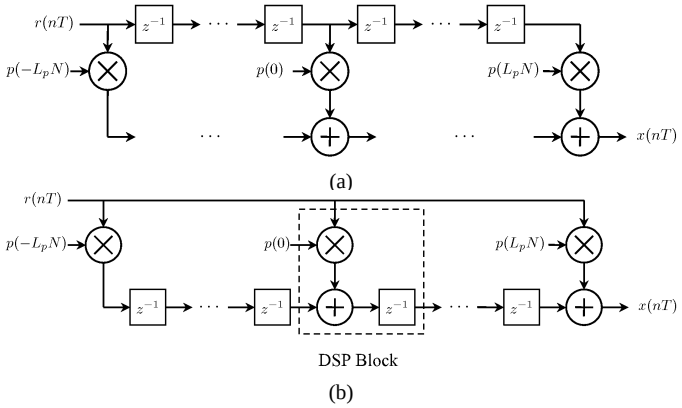

Fig. 2. The FIR filter structures examined in the fault injection experiments: (a) direct form 1 FIR filter; (b) transposed direct form 1 FIR filter.

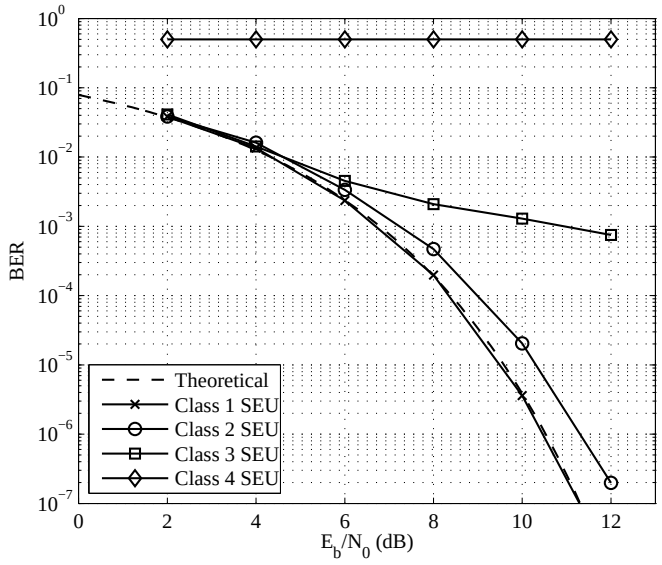

Fig. 3. BER plot showing representative samples from each of the four error classes from the 16-bit logic-based FIR filter with $\alpha = 1.0$.

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 SEU, a bit error rate curve was generated by producing $r(nT)$ given by (4) and processing it with the system defined by the corrupted configuration bit file.
5) For the non-catastrophic SEUs, the bit error rate curve produced by the previous step was compared to the curve for the system in the absence of upsets to estimate the performance loss at a bit error rate of 10^{-5}.

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

III. Experimental Results

Some representative examples of the bit error rate curves resulting from the fault-injection experiment are illustrated in Figure 3. The examples included in the figure are representative cases for what we consider to be four types of effects. We label these SEU categories "Class 1 SEU" through "Class 4 SEU." One of the contributions of this paper is to describe the location and function of these classes of SEUs in this system. Knowing which SEUs cause the more critical errors is a powerful tool for crafting a reduced-cost mitigation technique. A description of the SEU classes and their main causes is summarized as follows:

1) A Class 1 SEU causes almost no perturbation in the bit error rate performance of the matched filter detector. The measured loss is less than 0.1 dB. The SEUs in this class are those that alter the memory cells defining the low-order bits of the filter coefficients, the low-order bits of the outputs of the arithmetic units (i.e., the addition and multiplication blocks), etc.
2) A Class 2 SEU degrades the bit error rate performance in the same way an additional source of additive noise degrades performance. This effect can be thought of as either an implementation loss or, curiously, as a *noise figure*. Class 2 SEUs are those that impact the memory cells defining the middle-order bits of the filter coefficients, the middle-order bits of the outputs of the arithmetic units, etc.
3) A Class 3 SEU produces an unusably high bit error rate floor.[2] SEUs impacting the memory cells that define the high-order bits in the filter coefficients, the high-order bits in the outputs of the arithmetic units, etc. are the main causes of SEUs in this category. These SEUs are considered "catastrophic."
4) A Class 4 SEU produces a bit error rate of 1/2. These SEUs are also "catastrophic" and are caused by faults in the memory cells defining the clock distribution network, the global reset signal, the most significant bit (MSB) of the matched filter output, etc.

The number of SEUs in each class is a function of the properties of the filter coefficients (controlled in these experiments using the excess bandwidth parameter α), the number of bits used to quantify the filter coefficients, and the degree to which built-in units such as the dsp48 blocks are used. Graphical representations of the impact of all SEUs on the six designs used in the fault injection experiments are shown in Figures 4 – 9. In these figures, the number of SEUs corresponding to each point on the simulated BER curve is shown. These plots illustrate in dramatic fashion how the majority of the SEUs are Class 1 and Class 2 SEUs. Or, stated in another way, a relatively small percentage of the SEUs are catastrophic.

[2]Note that our simulations ran only long enough to estimate bit error rates greater than 10^{-6} with any useful reliability. It *could* be the case that many of the Class 2 SEUs really do have a bit error rate floor somewhere below 10^{-6}. A case could be made that these Class 2 SEUs should be Class 3 SEUs. Given the fact that most modern digital communication system use some form of error control coding and that any useful error correcting code can easily correct random errors at the rate of 10^{-6} or less, there is little merit in determining if such low bit error rate floors exist.

TABLE I
NUMBER OF SEUS CAUSING EACH CLASS OF EFFECT FOR SEVERAL DESIGNS.

Design	Class 1	Class 2	Class 3	Class 4	Total	Total Catastrophic
16b logic $\alpha = 1.0$	33,287	7,154	1,638	899	42,978	2,537 (5.90%)
16b logic $\alpha = 0.25$	21,072	44,205	2,908	1,022	69,207	3,930 (5.68%)
8b logic $\alpha = 1.0$	1,414	6,658	768	841	9,681	1,609 (16.62%)
8b logic $\alpha = 0.25$	1,508	13,547	1,816	908	17,779	2,724 (15.32%)
16b dsp48 $\alpha = 1.0$	20,514	7,031	867	1,118	29,530	1,985 (6.72%)
16b dsp48 $\alpha = 0.25$	7,395	30,606	1,263	1,031	40,295	2,294 (5.69%)

TABLE II
PERCENTAGE OF SEUS CAUSING CERTAIN SNR LOSSES AT BER OF 10^{-5}.

Design	> 0.1 dB	> 0.5 dB	> 1 dB	> 3 dB	> 6 dB
16b logic $\alpha = 1.0$	22.55%	16.31%	14.31%	11.20%	9.20%
16b logic $\alpha = 0.25$	69.55%	17.39%	14.36%	10.58%	9.08%
8b logic $\alpha = 1.0$	85.39%	51.93%	43.65%	33.33%	26.20%
8b logic $\alpha = 0.25$	91.52%	45.27%	37.19%	27.92%	24.11%
16b dsp48 $\alpha = 1.0$	30.53%	22.13%	20.18%	15.92%	12.05%
16b dsp48 $\alpha = 0.25$	81.65%	22.44%	18.38%	13.69%	10.92%

Numerical summaries are tabulated in Table I. An important observation is that the distribution of SEUs between Class 1 and Class 2 depends on the excess bandwidth α. This is due to the fact that when $\alpha = 1$, almost half of the filter coefficients are very close to 0. In fact, when 8-bit coefficients are used, these small filter coefficients are quantized to 0. The FPGA synthesis tool is smart enough to recognize that "multiplication by 0 followed by accumulation" is unnecessary and does not devote any resources to this operation. When $\alpha = 0.25$, most of the filter coefficients are sufficiently non-zero to survive quantization. Hence, the shortcut is not available to the synthesis tool and FPGA resources are devoted to the computation. This can be seen in the last column of Table I. The total number of SEUs (which is equal to the number of bits in the bit configuration file required to define the design) is larger for the $\alpha = 0.25$ design than for the corresponding $\alpha = 1$ design. What is interesting here is that the percentage of non-catastrophic SEUs remains approximately constant.

The SEUs may also be quantified by the implementation loss they cause. These results are summarized in Table II. These data define a cumulative distribution of the implementation loss[3] for each of the 6 designs. As an example, consider the designs using 16 bit filter coefficients with the filter structure of Figure 2 (a). Approximately 14% of all possible SEUs lead to an implementation loss in excess of 1 dB. In other words, 86% of all possible SEUs give an implementation loss less than 1 dB. The consequence of this observation is significant. If a 1 dB implementation loss is acceptable, only 14% of the SEUs need to be targeted for mitigation. This represents a huge potential savings in FPGA resources.

The situation is less dramatic for the design based on 8-bit filter coefficients. This is because all of the filter coefficient bits must be treated as "significant bits." As a result, the Class 2 SEUs are associated with higher implementation losses relative to the corresponding 16-bit designs and a larger percentage of the SEUs are Class 3 SEUs.

[3]Note that Class 3 and Class 4 SEUs have infinite implementation loss and are included in the percentages shown.

IV. SUMMARY AND CONCLUSIONS

We have shown that not all SEUs need to be targeted for mitigation in an FPGA design subject to SEUs. This desirable feature follows the fact that the figure of merit is bit error rate (rather than bit-level accuracy) and that the majority of the SEUs have the same effect as additive noise. The sections that must be protected from SEUs are the clock distribution networks, the MSBs of the arithmetic outputs, etc. Because not all SEUs need to be mitigated, much smaller designs are possible. This approach may substantially reduce the resources required to produce a reliable system.

It should be pointed out that we expect these conclusions to generalize only to feed-forward signal processing tasks. Recursive signal processing tasks, such as discrete-time phase lock loops (commonly used for carrier phase and symbol timing synchronization), will behave differently in the presence of SEUs and will require more aggressive mitigation techniques.

REFERENCES

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

[2] M. Caffrey, "A space-based reconfigurable radio," in *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA)*, T. P. Plaks and P. M. Athanas, Eds. CSREA Press, June 2002, pp. 49–53.

[3] M. Caffrey, P. Graham, M. Wirthlin, E. Johnson, and N. Rollins, "Single-event upsets in SRAM FPGA," in *Proceedings of the 5th Annual International Conference on Military and Aerospace Programmable Logic Devices (MAPLD)*, September 2002.

[4] E. Johnson, M. J. Wirthlin, and M. Caffrey, "Single-event upset simulation on an FPGA," in *Proceedings of the International Conference on Engineering of Reconfigurable Systems and Algorithms (ERSA)*, T. P. Plaks and P. M. Athanas, Eds. CSREA Press, June 2002, pp. 68–73.

[5] C. Carmichael, M. Caffrey, and A. Salazar, "Correcting single-event upsets through Virtex partial configuration," Xilinx Corporation, Tech. Rep., June 1, 2000, xAPP216 (v1.0).

[6] C. Carmichael, "Triple module redundancy design techniques for Virtex FPGAs," Xilinx Corporation, Tech. Rep., November 1, 2001, xAPP197 (v1.0).

[7] M. Rice, *Digital Communications: A Discrete-Time Approach*, 1st ed. New Jersey: Pearson Prentice Hall, 2009.

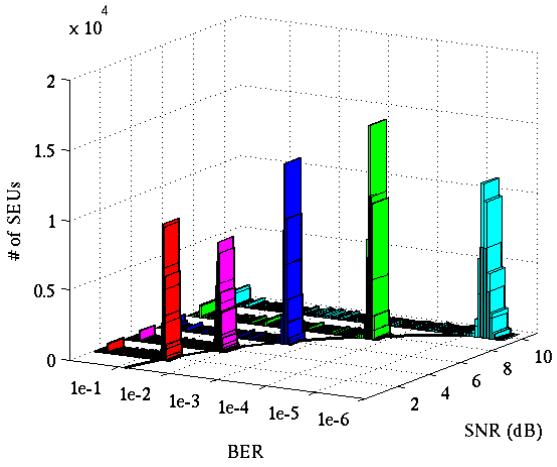

Fig. 4. BER plot for the 16-bit logic-based FIR filter with $\alpha = 1.0$.

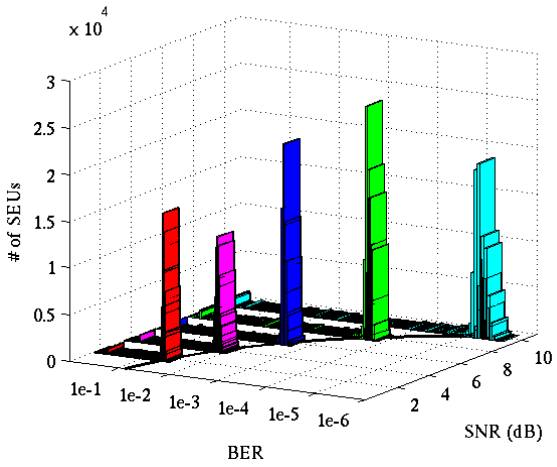

Fig. 5. BER plot for the 16-bit logic-based FIR filter with $\alpha = 0.25$.

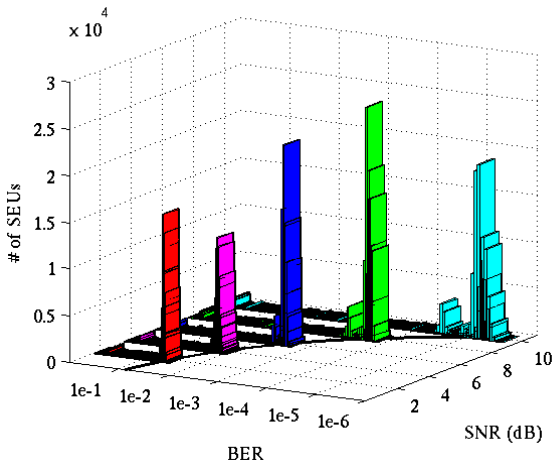

Fig. 6. BER plot for the 8-bit logic-based FIR filter with $\alpha = 1.0$.

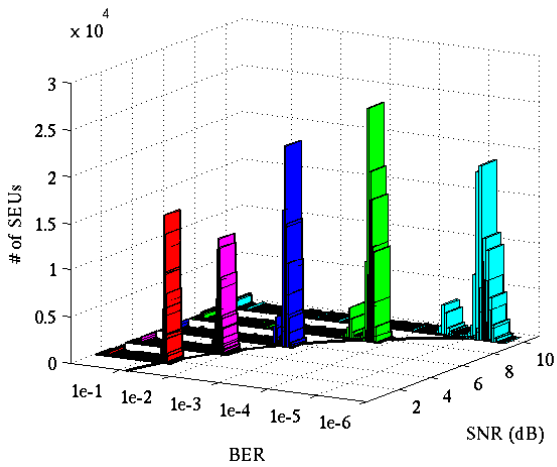

Fig. 7. BER plot for the 8-bit logic-based FIR filter with $\alpha = 0.25$.

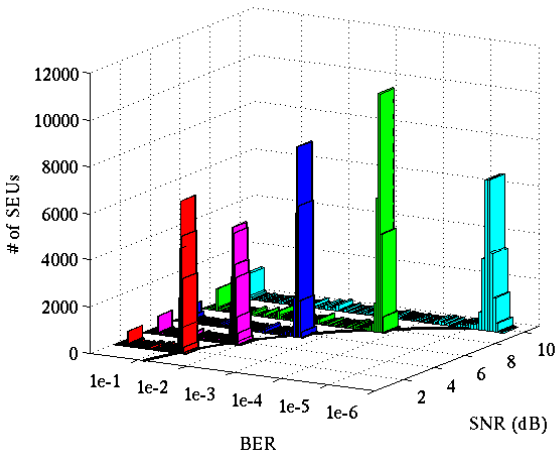

Fig. 8. BER plot for the 16-bit DSP48-based FIR filter with $\alpha = 1.0$.

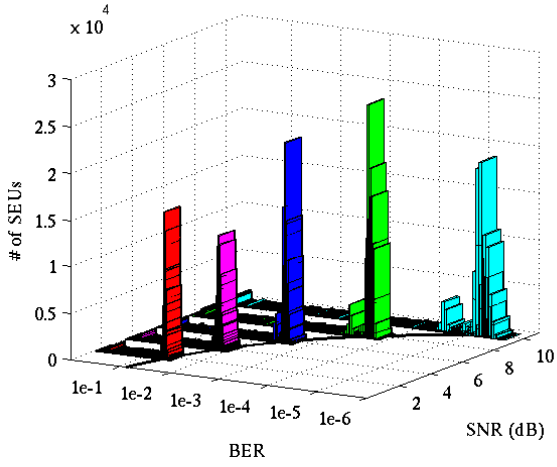

Fig. 9. BER plot for the 16-bit DSP48-based FIR filter with $\alpha = 0.25$.