# A Field Analysis of System-level Effects of Soft Errors Occurring in Microprocessors used in Information Systems

Syed Z. Shazli, Mohammed Abdul-Aziz, Mehdi B. Tahoori, David R. Kaeli
*Department of Electrical and Computer Engineering,*
*Northeastern University, Boston MA*
*Emails:{sshazli,mabdul,mtahoori,kaeli}@ece.neu.edu*

## Abstract

*Soft errors due to alpha and cosmic particles are a growing reliability threat to information systems. In this work, a methodology is developed to analyze the effects of single event upsets (SEU) and obtain FIT rates for commercial microprocessors in live information systems. Our methodology is based on data collected from error logs and error traces of the information systems present globally in the field. We also compare the system effects of errors that are suspected to be due to SEUs as compared with non-SEU errors. Soft errors are further localized within specific microprocessor resources with the assistance of the machine check architecture. The analyzed field data represents a world-wide population of microprocessors installed in the field. In total, several thousands systems and thirty-six months of field data were analyzed. The methodology used in carrying out this field analysis is discussed in detail and results are presented.*

## 1. Introduction

Information systems are employed in an increasing number of application areas, motivating research activity focused on reducing the time–to–market, improving performance, reducing power consumption and increasing reliability. This latter aspect has assumed an increasingly important role, especially when dealing with critical applications. As device geometries continue to shrink, microprocessors will experience an increasing number of soft errors. These bit flips can temporarily corrupt the data being processed. As a consequence, future designs will need to be able to detect permanent hardware failures and recover from transient errors caused by soft errors [9][16].

Soft errors are occasional malfunctions of the hardware that are not reproducible. Single Event Upsets (SEUs) that cause soft errors are generated by cosmic particles, energetic neutrons and alpha particles hitting the surface of silicon devices [1]. As feature sizes shrink, the amount of charge per device decreases, and so a particle strike is much more likely to cause an error. Particles of lower energy, which are more abundant than high energy particles, will generate sufficient charge to cause a soft error. In the absence of error correction schemes, the system error rate will grow in direct proportion to the number of bits on the chip. With the exponential increase in transistor counts (as predicted by Moore's Law), we expect to see a similar increase in error rates for unprotected systems [8].

Soft errors are emerging as a significant obstacle to increasing microprocessor transistor count in future process technologies. Although soft error rates of individual transistors are not projected to rise, incorporating more transistors into the same device real estate makes a device more likely to encounter a fault. As a result, it is expected that maintaining microprocessor error rates at acceptable levels will require specific design changes [2].

A key requirement of present day information systems is availability. Sometimes transient errors take the system to an invalid state. Large systems may take a considerable amount of time to recover. Hence, these errors can considerably impact system availability [6].

In this paper, we focus on *soft error rate* (SER) estimation of microprocessors used in information systems by analyzing actual field data. This work is done based on our collaboration with a major industrial information computing system manufacturer. This work complements the work done in [7], which analyzed field data for computing SER in FPGA-based designs. Since the internal structure of the microprocessor is not observable at the system level,


1-4244-4203-0/08/$20.00 ©2008 IEEE

determining soft error rates using field data for microprocessors is more challenging compared to performing the same study for FPGA-based designs. We present a case study of failure rate data for commonly used 32-bit server processors in different information systems. We have further analyzed the exact locations of SEUs within the microprocessor and compared results against our previous analytical models. In addition to computing the SER for the microprocessors used in a commercial information system, we also consider the system effects of non-SEU events such as permanent faults and software failures. A comparison on the reliability impact of SEU and non-SEU events has been performed. It is observed that the SEUs occurring within the microprocessors are mostly affecting the tag array of the caches. As tags are only parity protected in most microprocessors, the SER is dependent on the size of the cache. This study is based on field data collected from several thousands of live commercial information systems over a three-year period.

The remainder of this paper is organized as follows. In Section 2, an overview of the microprocessor, along with its usage in the information system, is presented. A brief review of soft error rate estimation for microprocessors is provided in Section 3. Section 4 details the methodology used in this study to estimate the soft error rate for the microprocessor used in the information system. Some case studies highlighting the methodology used are presented in Section 5. Section 6 covers the analysis of possible SEUs occurring in the system along with computation of FIT rates. We have compared the effects of SEU and non-SEU related incidents in Section 7. Analysis of these results is presented in Section 8. Finally, Section 9 concludes the paper.

## 2. Overview of the information systems

The class of information system under consideration in this study is a high-availability system equipped with multiple microprocessors. These information systems have hundreds of terabytes of drive capacity and can serve more than a hundred connected hosts.

The internal bussing architecture provides for a high degree of redundancy so that a failure in any component on a link does not disconnect other components from the system. However, simultaneous failures in multiple components can lead to events which impact reliability and potentially reduce availability. We call these *Reliability Impact Events* (RIEs). Such events may cause sudden interruptions in service and potentially result in data unavailability. Our analysis is based on studying such events in the error logs of live systems.

One of the components that interconnect to the buses is the *Logical Unit Module* (LUM). Each LUM contains one or two single-core state-of-the-art IA-32 microprocessors (the so-called "server processors"), depending on the particular system. The microprocessors used in these LUMs act as the main control center for the system. They are responsible for processing I/O requests arriving from the attached host(s).

On powering up the system, each microprocessor boots up an operating system. The operating system manages all functions of the information system and supports some basic configuration operations. A load balancing utility running on the host schedules jobs between the two microprocessors. If an RIE appears in one of the microprocessors (or LUM), the utility transfers all the jobs to the other microprocessor (or LUM). If the second microprocessor is already heavily utilized, the system may crash. The LUM is responsible for all data buffering and caching, though performs no processing on the stored data. Since data is heavily protected with ECC, soft errors in LUMs cannot affect the data items and cause *silent data corruption* (SDC). Therefore, RIEs (availability degradation) are the main impact of SEUs in these systems.

The microprocessor under study supports a *Machine Check Architecture* (MCA) as described in [6] [17] and [20]. Issues related to the design of MCAs have been studied in earlier reliability work targeting server systems [14]. The MCA allows the operating system to detect, signal, and record information about selected machine fault conditions. Some of those faults are correctable, while others are uncorrectable (i.e., only detectable).

The machine check mechanism is intended to enable system providers and system software developers to diagnose, isolate, and understand microprocessor failures. It is also intended to enable system recovery mechanisms to be employed. By architecting the MCA, the software and operating system community can count on the same level of support in future versions of the microprocessor. Further details about the MCA are discussed in Section 4. In the following, we give details of the two types of systems under study.

**System Type A** - In these systems, there are two server microprocessors on each LUM and two LUMs per system, for a total of four microprocessors per system. Two levels of on-chip caching are available on these microprocessors. The L1 data cache is parity protected, while the L2 cache has ECC protection on

data and parity protection on tags. The processors used in System A, are superscalar and have a 20-stage pipeline.

**System Type B** – Similar to Systems A, these information systems have two LUMs in each system. We break down the machines included in Systems B into two subclasses (Systems B1 and Systems B2). While there are two processors per LUM in System B1, System B2 contains only a single processor per LUM. All other aspects of Systems B1 and B2 are exactly the same. As System B is a newer generation system than System A, it uses a more recent version of the server processor. The drive capacity of System B is almost twice as that in system A. In the microprocessors used in System B, the size of L1 and L2 cache is twice the size of the corresponding caches in microprocessors of System A. Similar to microprocessors used in System A, the L1 data cache is parity protected, while the L2 cache has ECC protection on data and parity protection on tags. The population of System B in the field is substantially smaller than System A. The processors used in Systems B, employ a 20-stage pipeline and run at twice the speed of processors used in Systems A. These are built using a newer technology node.

Our analysis is based on all running systems in the field of these two types for a period of more than three years, though System B have only been in the field for two years.

## 3. Soft Error Rate estimation in microprocessors

A prior software-simulated fault injection study [1] used an RTL model of the PicoJava-II microprocessor to determine the soft error sensitivity of logic blocks within the design. The soft error sensitivity (SES) metric used in that work was computed as the probability of whether a soft error within a given logic block will cause the microprocessor to enter an incorrect architectural state.

In [2], the authors defined the term architectural vulnerability factor (AVF) to be the probability that a fault in a micro-architectural structure will cause an error in program output. They used a performance simulator of the Itanium II microarchitecture to determine the AVF for each structure within their simulated microarchitecture.

The effects of soft errors on an out-of-order, superscalar Alpha-like microprocessor core were characterized in [3]. The fault model used in that work simulated single bit flips in sequential state elements within the design, and analyzed the failure modes exhibited in simulation.

A similar analysis of the effects of soft errors occurring in both sequential state elements and combinatorial logic on a DLX microprocessor model was presented in [4]. The error manifestation rates of different architectural blocks were reported.

A thorough micro-architectural analysis of the effects of soft errors on a production-level Verilog implementation of an ARM926EJ-S core was carried out in [5]. The authors examined the propagation of faults occurring in both sequential state elements and combinatorial logic. They identified a number of critical differences in error propagation behavior of soft errors occurring in logic gates versus state-based elements.

The susceptibility of commodity operating systems running on IA-32 and IA-64 microprocessors to soft errors was investigated in [6]. The results indicated that with improved microprocessor support like the MCA, and a little application knowledge, few of the detected soft errors needed to result in fatal system errors, especially because many of these bit flips would be overwritten in the future.

Radiation testing has been done on microprocessors in [18][19]. The impact of semiconductor technology scaling on neutron induced soft errors has been studied using accelerated testing in [18]. In earlier experiments in [19], the effect of soft errors at various altitudes has been reported. Radiation testing was done and it was observed that soft error fails increase by several orders of magnitude with increasing altitudes.

## 4. Methodology

### 4.1 SEUs in the field

Field data analysis is traditionally used in industry to evaluate system failure rates, monitor system reliability, and conduct long-term trend analysis to identify the need for future design changes [11][15]. The value of any field study is highly dependent upon the quality and completeness of the event logs. If the information is missing, incomplete, or difficult to interpret, our ability to analyze the data is greatly diminished. Associating an error event with an SEU in the microprocessor is significantly more complex compared to doing the same for FPGAs [7]. In the case of FPGAs, the configuration bits can be read out and checked for bit-flips. Any user bit can be written and read back out to verify if the bit flip was transient or stuck-at. This is not easy to do in the case of microprocessors, since registers and flip-flops are typically not observable (unless trace arrays are provided [13]).

As a figure of merit we consider *Failures in Time* (FITs), which is defined as the number of failures per

billion operating hours. The aim of our study is to calculate the FIT rate due to soft errors for the microprocessors used in the LUMs, based only on the SEUs observed in the field. Failure logs were obtained from the repository of field data available from the manufacturer.

In the information systems studied in this work, we limit our focus on microprocessor RIEs that did not result in LUMs being replaced in the system, corresponding to one-time RIEs. We therefore investigated all of the possible causes of RIEs. The methodology used to identify possible SEUs is shown in Fig 1 and is outlined below:

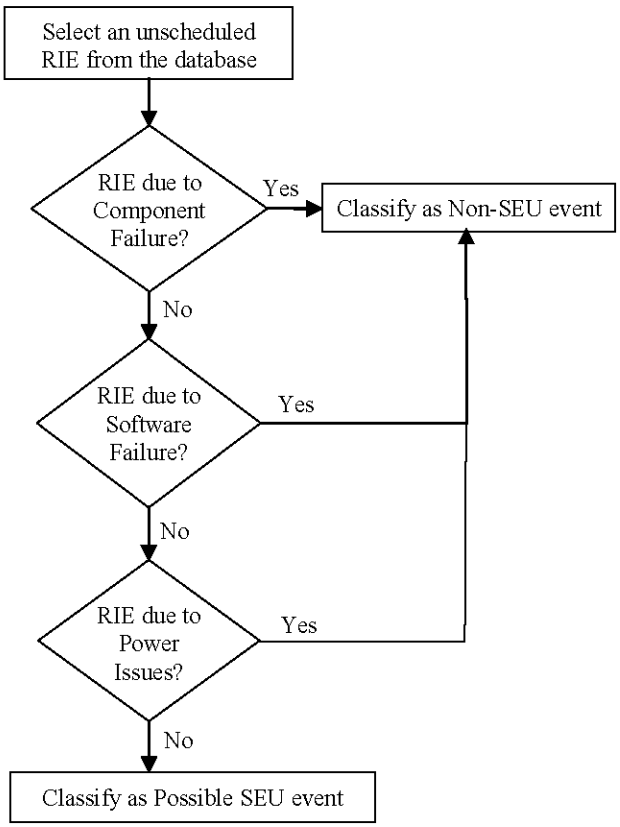



**Fig. 1 Methodology to Identify Possible SEUs**

Initially, all unscheduled RIEs occurring in the microprocessors contained in the LUMs were analyzed. A large number of unscheduled RIEs were identified. These can occur for one of several possible reasons:

a) Component failure
b) Software-based errors
c) Power issues
d) Radiation-induced errors.

The next step was to isolate those cases that could have occurred because of a soft error. We call these *Possible* soft errors.

Errors that were confirmed to have occurred inside the microprocessor using error logs and traces were labeled as *Probable* soft errors and are denoted by $SEU_{CPU}$. Most of these cases resulted in Machine Check exceptions. In some cases, we were able to identify the register in which a bit flip had occurred, with the help of error traces.

In some cases invalid memory addresses were generated by the microprocessor and resulted in a microprocessor RIE. If an RIE could not be linked to faulty software, and the traces did not point to a specific register in which the error occurred, we refer to these cases as $SEU_{INV\text{-}MEM}$. Since invalid addresses may still be generated because of a software bug, we do not include them in the *Probable* soft error bucket. Instead, we construct a *Potential* soft error bucket and include such cases in this category.

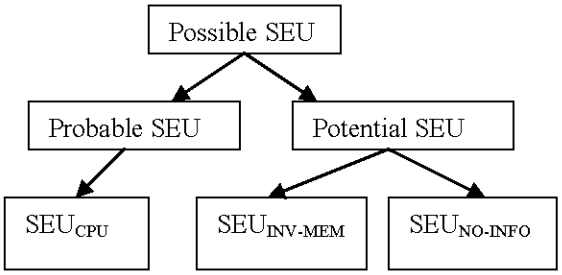



**Fig. 2 Classification of SEU events**

The *Potential* soft error bucket also contains those cases of isolated RIEs in which the logs did not point to a specific fault in hardware or software. The error could not be confirmed to have occurred in the microprocessor or be attributed to an invalid memory address. Most of these errors were classified as *Potential* because some of the logs that were studied were incomplete. However, since these were isolated cases out of the total number of RIEs, they were potential candidates for SEUs. We include these cases in a category called $SEU_{no\text{-}info}$. Our classification taxonomy is shown in Fig 2.

The following approach was taken to identify these cases:

i) Note the time and date of the RIE.
ii) Search the events in the microprocessor logs, for that incident.
iii) If the events preceding the incident can be classified as reasons a, b or c above, do not include them in the *Possible* SEU bucket.

Using the above criteria, we obtained a bucket containing *Possible* soft errors that had a significant number of incidents. Note that these are individual instances of RIEs that could not be linked to a hardware component, power supply issue, or corrupted software.

### 4.2 Localizing SEUs within the microprocessor

The Probable SEUs occurring inside the microprocessor were further investigated with the help of the Machine Check Architecture (MCA) information available to help to isolate just exactly where in the microprocessor they occurred. The machine check architecture is an internal architecture subsystem which detects and captures errors occurring within the microprocessor logic [17][20]. It handles five main subsystems.

- External Bus Logic
- Back-side Bus Logic
- Cache Unit
- Translation Look aside Buffer
- Instruction Fetch Unit

The machine check exception handler is the only OS-level tool which guarantees that an error occurred inside the processor. The exception handler returns a 64-bit error code. The error code helps in finding out where in the cache hierarchy the error occurred. Bits 0 to 15 contain the machine check error code, and bits 16 to 31 contain the model-specific error code. There are some reserved and informational bits and also some flags. The format of the 16-bit error code is as follows: `0000 0001 RRRR TTLL`. The description of the fields is as follows.

- The 2-bit TT sub-field indicates the type of transaction: data (00), instruction (01), or generic (10). A generic type is reported if the microprocessor is unable to determine the type of the transaction.
- The 2-bit LL sub-field indicates the level in the memory hierarchy where the error occurred: level-0 (00), level-1 (01), level-2(10), or generic (11). Again, the generic type is reported when the microprocessor cannot determine the hierarchy level.
- The 4-bit RRRR sub-field indicates the type of the action associated with the error. Actions include read and write operations, prefetches, cache evictions, and snoops.

## 5. Case studies

A significant number of RIEs were received due to invalid addresses being generated by the processor. In general, invalid memory accesses are typically caused by software faults. However, if the logs do not contain any further memory access violations while using the same software, we have included these incidents in the *Potential* SEU bucket. Similarly, some RIEs may have occurred because of power spikes. However, if the logs did not show any power related problems on the system before or after the event, we included the incident in the *Potential* SEU bucket. In this section, we will present four examples to show how an isolated RIE can point to a soft error.

### 5.1 A probable SEU in the Instruction Pointer

We first examine an RIE that occurred in the microprocessor because of an invalid memory address. With the help of error traces, we were able to identify the events leading up to the incident. The following is the sequence of events prior to the incident.

- Function `ABC` was called just before the RIE occurred.
- Before calling `ABC`, the return address was pushed on the stack. The calling function was in the range `0xa5be8xxx – 0xa5beexxx`.
- The value of the Instruction Pointer (IP) at the time of RIE was `0xa7be8a8e`. This was an invalid address that resulted in the incident being signaled.
- Changing the IP from `0xa7be8a8e` to `0xa5be8a8e`, which lies within the range of the calling function, we can see that an invalid value was popped off the stack on the return from the function `ABC`. The true value differed from the actual value by one bit (the 7$^{th}$ bit from the left), and this was not identified to be a software bug. The incidence was included in the *probable* SEU bucket because the error was confirmed to have occurred inside the processor.

### 5.2 A bit flip in the address of a function

In another incident, an attempt was made to access an invalid memory address that resulted in a RIE. Stack traces were uploaded and the logs pointed to the following events prior to the incident:

- The last call before the incident references the address `0xf9a96652`. This is an actual function pointer, so this seems to be a valid value. Looking at the lines that should have been executed we find:

```
FunctionABC:
0xf9a96652  55     push ebp
```

The value at this address translates to `push ebp`. However, the value on the stack is not `ebp`.

- The address where the system incurred this error was `0xf9a96258`, which is not in the function.
- If we look at the instructions preceding the instruction where the system halted, we find:

```
0xf9a96252  53    push ebx
0xf9a96253  183b  sbb [ebx],bh
0xf9a96255  55    push ebp
0xf9a96256  d476  aam …
0xf9a96258  0884c90f851304
      or[ecx+ecx*8+0x413850f],al
```

The values in `ebx` and `ebp` match exactly the values on the stack.

The last instruction at `0xf9a96258` referenced an invalid pointer that resulted in an RIE.

- We can see here that instead of going to `0xf9a96652`, the microprocessor went to the address `0xf9a96252`, which is a difference of 1 bit (the 11th bit from the right). Since the error is confirmed to have occurred inside the processor, we include this incident in the set of *Probable* SEUs.

### 5.3 A bit flip in the instruction cache

For a particular RIE, a machine check exception was generated and the error code returned was `0xBE00000020010152`. When we inspect the last 16 bits (`0000 0001 0101 0010`), we see that the 2 rightmost bits (10) denote that the error occurred in the L2 cache. The next 2 bits (00) indicate that the error was in the instruction cache. Code 0101 translates to a read error. Hence, we can conclude that a parity error occurred in the L2 instruction cache.

### 5.4 A potential SEU due to an invalid memory address

In one instance of an RIE, an invalid memory address was generated which led to a memory access violation. A software bug is generally the most probable cause of invalid memory accesses. However, in this RIE incident, there was no subsequent memory violation, even though the software running on the machine was not changed or upgraded. Since, the error was not reproducible and did not occur because of a specific software or hardware issue, we included it in the *Potential* SEU bucket in the $SEU_{INV\text{-}MEM}$ category.

## 6. Analysis of possible SEUs

Our study is based on field data obtained over a three-year period. This comprehensive field analysis was not limited in scope to one particular geographic location; data was collected from all functioning systems spanning installations from all across the globe. In total, several thousand systems and thirty-six months of field data were analyzed.

We initially looked at all unscheduled RIEs occurring in the microprocessors used in the LUMs. By looking at error logs and error traces, we tried to determine the root cause of each incident. Those incidents that resulted from a Machine Check exception were further investigated to identify the location in the microprocessor where the error occurred. The exception handler returned a 64-bit error code. This error code was decoded to identify the failing unit inside the microprocessor. We present our findings in the following sections.

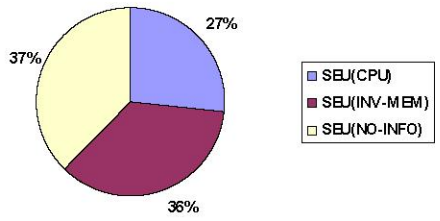
### 6.1 SEU distribution in the systems

There were a significant number of unscheduled RIEs that occurred in these systems. Over a hundred of these were classified as *Possible* SEUs. Figure 3 shows the statistics of *Possible* SEU events occurring in the two systems.

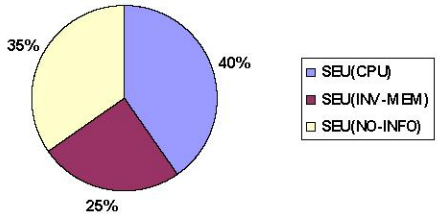
It was observed that some of the RIEs were due to events occurring inside the CPU ($SEU_{CPU}$). Other events were due to invalid memory addresses generated by the microprocessor ($SEU_{INV\text{-}MEM}$). Since these invalid addresses resulted in a single instance of an RIE and were not linked to faulty software, they were included in the *Potential* soft error bucket. A similar number of indeterminate cases were found where there was an isolated instance of an RIE, though it could not be linked to faulty hardware or software. In some cases, there was insufficient information in the logs, but since these were single instances of an RIE, and they were not linked to any software or hardware failure, they were included in the *Potential* SEU bucket. We group them as $SEU_{NO\text{-}INFO}$ in Fig. 3.

### 6.2 SEU localization within the microprocessor

When an SEU is confirmed to have occurred inside the microprocessor, we classify it as a *Probable* SEU. For systems of type A we analyzed a statistically significant number of probable SEUs and identified the exact location of SEUs occurring within the microprocessor using the Machine Check Architecture (MCA) information [17]. MCA identifies errors occurring in the cache hierarchy as well as those occurring in other structures as described in Section 4.2. Additionally, some invalid memory accesses which were traced back to bit flips in the processor registers were also included in the probable SEU list. As mentioned in Section 2, the L1 cache in the microprocessor under study has parity protection. The L2 cache is an 8-way set associative cache with ECC protection on the data array and parity protection on tag array.




(a) System A




(b) System B

**Fig. 3 Distribution of *Possible* SEUs in the two systems**

As mentioned in [12], ECC protection reduces the effective vulnerability of the L2 data array to zero and the primary source of L2 vulnerability is the tag array. It was observed that the L2 tag vulnerability is greater than the vulnerability of L1 data and Instruction cache for most of the SPEC benchmarks when a similar processor and memory organization was considered.

To further investigate the vulnerability of the L2 tag addresses, the L2 tag vulnerability was profiled in terms of pseudo-hit vulnerability, replacement vulnerability, multi-hit vulnerability, and status vulnerability in [12]. The results show that replacement vulnerability makes up almost 85% of the total tag vulnerability. This is due to the fact that tag-addresses become more susceptible once the first write occurs in the block and remain susceptible to SEUs until the block is replaced or flushed to lower levels of memory.

The field analysis carried out in this work agrees with the findings presented in [12]. With the help of the MCA given in [17], it was observed that only 12% of the errors were found to have occurred in the L1 cache. This is primarily due to its small size. Almost 80% of the *probable* SEUs occurred on tag bits of the L2 cache. Additionally, there were some parity errors in the interconnect buses (which are only parity protected). There were no errors found in the other structures using the MCA. The results are presented in Table 1.

**Table 1: Statistics for Machine Checks in Systems A**

| | |
|---|---|
| Parity on L2 Tag | 80% |
| Data Parity on L1 Cache | 12% |
| Interconnects | 8% |

### 6.3 Calculating FIT rates due to SEUs

The FIT rates were calculated separately for *Probable* SEUs and all *Possible* SEUs in the two types of systems. They were obtained by dividing the number of SEUs by the total run time of all systems. Run times were computed using the following formula:

$$runningtime = \sum_i system_i \times time\_in\_field_i$$

for $system_i$ in the field for $time\_in\_field_i$.

**Table 2 Relative FIT rates**

| | Relative FIT (Probable SEU) | Relative FIT (Possible SEU) |
|---|---|---|
| System A | 1 | 3.68 |
| System B | 3.24 | 8.03 |

In this study, more than half a billion system-hours of information systems in the field were analyzed. System A has significantly higher systems-hours than system B. Table 2 shows the relative FIT rates (to the *Probable* SEU FIT rate of system A) for these systems based on *Probable* and *Possible* SEUs (i.e., the lower and upper bounds). Due to extreme sensitivity of such data, we have only reported the relative values instead of the actual FIT rates in this paper.

## 7. Comparison of SEU and non-SEU events

We further break down the machines included in Systems B into two subclasses (Systems B1 and Systems B2). While there are two processors per LUM in System B1, System B2 contains only a single processor per LUM. The processors used are exactly the same in Systems B1 and B2. We studied the relative frequency of *Possible* SEUs compared to other causes of RIEs for these systems. The results for each system type as well as total RIE cases are presented in Table 3. The results for each column (representing each system type) are normalized to the number of

*Possible* SEUs for that system. The graphical representation of this data is shown in Fig. 4.

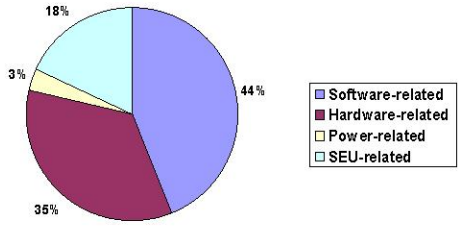
**Table 3 Characterizing the cause of RIEs**

| Source of RIE | System A | System B1 | System B2 | Total |
|---|---|---|---|---|
| Hardware-related | 1.91 | 2.27 | 7.25 | 2.19 |
| Power-related | 0.18 | 0.19 | 0.5 | 0.19 |
| Software-related | 2.41 | 4.44 | 18.12 | 3.48 |
| *SEU-related* | *1.0* | *1.0* | *1.0* | *1.0* |

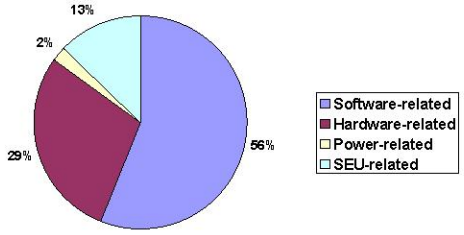
RIEs can occur due to several reasons. The first three entries in Table 3 are Non-SEU related, whereas the last entry is SEU-related. These categories can be explained as follows.

- *Hardware-related (permanent) failures:* These failures are permanent faults attributable to a faulty hardware component. For instance, the hardware of the microprocessor may itself be defective. This is observed when there are multiple machine checks in a short period of time. Sometimes faults on other components connected with the microprocessor (associated components) may cause RIEs.
- *Power-related failures:* Although the information system has spare power supplies, power spikes and failures can result in RIEs. If the events preceding the RIE point to unstable power supply, the incident is classified as a Power-related failure.
- *Software-related failures:* A large percentage of the incidents occurred because of software errors. This includes the software running on the host (e.g., the load balancing software) and the operating system running on the microprocessor. RIEs can also occur when the system is improperly configured at installation time or during an upgrade.
- *SEU-related failures:* These temporary failures correspond to all *Possible* SEUs. This category contains all RIEs due to *Probable* and *Potential* SEUs.

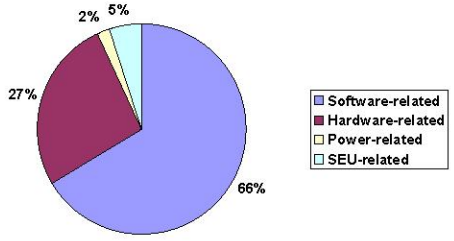
Hardware-related failures are permanent, whereas the other three types of failures are temporary. It can be observed that power-related failures constitute a very small percentage of the total RIEs. It needs to be mentioned that although software-related failures correspond to the largest percentage of RIEs, they can be fixed using error traces and error logs.



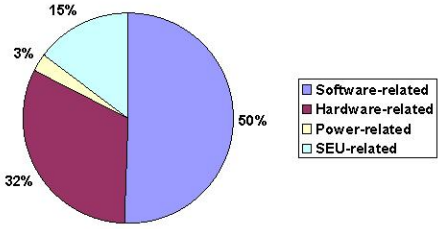

(a) System A




(b) System B1




(c) System B2




(d) All systems combined

**Fig. 4 Distribution of various causes of RIEs in different systems**

Every RIE is characterized by a high, medium or low severity, by field engineers, depending upon its availability impact and the availability requirements of

the particular system in the field. The results are graphically depicted in Fig. 5. Each bar in the figure represents the severity levels of various causes of RIE, as described in Sec.6.3. The number in the brackets is the average severity level. The severity values ranges from 1 (low) to 3 (high).

It can be observed from Fig. 5 that severity of various sources of failures is system-dependent. For newer systems (systems of type B1 and B2), software-related failures are the most severe. For more stable systems (type B2) with more developed software, hardware-related failures are most severe. On average (see Figure 4.d), hardware-related failures are more severe than transient failures. The average level of severity of SEU-related RIEs is slightly smaller as compared to other transient causes. This is because SEUs can result in isolated instances of RIEs and that the machine works normally after RIEs occur (i.e., no system errors are a direct result of SEUs occurring). In summary, SEUs are associated with the least severe RIEs in current systems. However, as the frequency of soft errors exponentially increases with each new technology generation, the severity of SEU-related failures is expected to increase accordingly.

## 8. Analysis of Results

The FIT rate analysis of the microprocessor used in these systems, as presented in Table 2, suggests that the FIT rates of the microprocessors in System B are significantly (2-4 times) higher than System A.

Studying the architectures of Systems A and B, as presented in Section 2, shows that the on-chip cache of System B is twice the size of the cache in System A. Although the computed FIT rates are limited to particular RIE events and analysis of error logs, a comparative analysis can factor out common sources of error in the analysis. This data suggests that the system-level FIT rate of server microprocessors in this study has a direct relationship to the size of on-chip cache memory.

Moreover, the distribution of SEU and non-SEU related failures, as presented in Fig. 4, shows that the percentage of SEU-related failures in systems of type A and B1 are almost the same and almost twice of that in type B2. The structures of these systems, as presented in Sec. 2, show that systems of type A and B1 have the same number of microprocessors per system (4) and twice of that in type B2 (2). Therefore, this data suggests that the percentage of SEU-related failures in the system is proportional to the number of microprocessors in the system, as expected.

It is important to note here, that in many instances complete logs were not uploaded at the time when RIE occurred. This prevented us from finding the root cause of those events. In some cases, the customer was not interested in an isolated RIE instance for example, and did not ask for finding out the root cause. Additionally, since the process of uploading the logs in the database is semi-manual, there are some human errors in the process. If the process is fully automated, the logs would be much more complete and hence easier to analyze.

## 9. Summary

In this study, we focused on the SEUs occurring in state-of-the-art microprocessors used in high performance information systems. Only system-level effects of soft errors were considered. These incidents, termed as Reliability Impact Events (RIE), can take several tens of minutes or more to service on large systems and can affect availability considerably. It is important to note that the issue under consideration is data un-availability rather than data corruption.

We looked into systems equipped with a different number microprocessors per system and different on-chip cache sizes. We divided the incidents of RIEs into two main categories: 1) incidents possibly due to SEUs and 2) incidents due to other permanent and temporary faults (non-SEUs). We further localized the SEUs to those occurring in the microprocessor (*Probable* SEUs) and those observed in memory (*Potential* SEUs). The frequencies of occurrence of various SEUs and non-SEUs were reported. We also carried out a severity analysis of various categories of errors. We found that the error trends found in particular memory structures inside the microprocessor agree with results obtained in earlier work using analytical models. Also, the size of on-chip cache memory has a direct impact on the system-level FIT rate.

## References

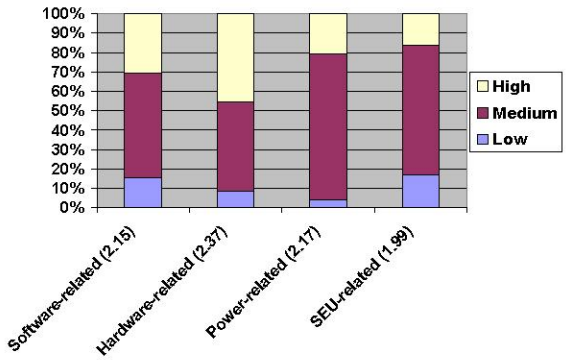

[1] S. Kim et. al. "Soft Error Sensitivity Characterization for Microprocessor Dependability Enhancement Strategy," *Proc. International. Conference on Dependable Systems and Networks*, pp. 416-425, Jun 2002.

[2] S.S. Mukherjee et. al. "A Systematic Methodology to Compute the Architectural Vulnerability Factors for a High-Performance Microprocessor," *Proc. International. Symposium on Micro-architecture (MICRO-36)*, pp. 29-40, 2003.

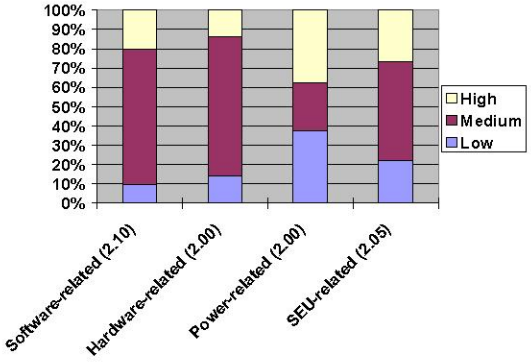
[3] N. Wang et. al., "Characterizing the effect of Transient Faults on a High Performance Processor Pipeline", *in Proc. International Conference on Dependable Systems and Networks,* 2004.

[4] G. Saggese et. al., "An Experimental Study of Soft Errors in Microprocessors", *IEEE MICRO*, pp. 30-49, Nov-Dec 2005.

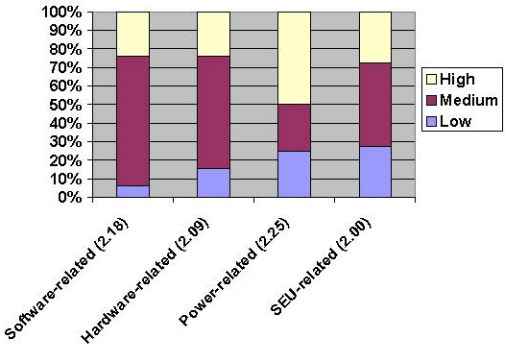
[5] J. Blome et. al., "A Microarchitectural Analysis of Soft Error Propagation in a Production-Level Embedded Microprocessor", *Proc. First Workshop on Architectural Reliability (WAR),* Nov 2005.



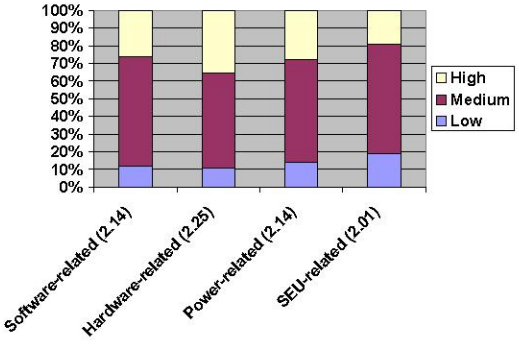

(a) System A




(b) System B1




(c) System B2




(d) All systems combined

**Fig. 5 Severity of RIEs due to various causes in different systems**


[6] A. Messer et. al., "Susceptibility of Commodity Systems and Software to Memory Soft Errors", *IEEE Transaction on Computers,* Vol. 53, No. 12, pp 1557, Dec 2004.

[7] B. Mullins et. al. "Case Study: Soft Error Rate Analysis in Storage Systems", *Proc. IEEE VLSI Test Symposium,* Apr 2007.

[8] R.C. Baumann, "Radiation induced Soft Errors in Advanced Semiconductor Technologies", *IEEE Transactions on Device and Materials Reliability,* Vol. 5, No. 3, pp 305-316, Sep 2005.

[9] C. Bolchini et. al. "A model of soft error effects in generic IP processors", *Proc. IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems,* 2005.

[10] J. Hennessy et. al., "Computer Architecture: A Quantitative Approach", Fourth Edition, *Morgan Kauffmann Publishers*, 2006.

[11] M.F. Buckley et. al. "VAX/VMS event monitoring and analysis" *Proc. International Symposium on Fault Tolerant Computing,* pp 414-423, Jun 1995.

[12] H. Asadi et. al. "Vulnerability Analysis of L2 Cache Elements to Single Event Upsets", *Proc. Design and Test Conference in Europe,* pp. 1276-1281, Mar 2006.

[13] B. W. Curran et. al. "IBM Enterprise System/9000 Type 9121 System Controller and Memory Subsystem Design", *IBM Journal of Research and Development,* Vol. 35, No. 3, pp 357-366, Sep 1991.

[14] K. Johnson et. al. "The Use of Machine Check Architecture (MCA) to Perform Reliability Studies on Servers", *Proc. Quality and Productivity Research Conference*, Austin, Texas, May 2001.

[15] R. Velazco et. al. "How to Characterize the Problem of SEU in processors & Representative Errors Observed on Flight", *Proc. IEEE International On-Line Testing Symposium*, pp. 303-308, 2005.

[16] C. Weaver et. al. "Techniques to reduce the soft error rate of a high-performance microprocessor", *Proc. Annual International Symposium on Computer Architecture*, pp. 264-275, 2004.

[17] AMD64 Architecture Programmer's Manual Volume 2: System Programming, Advanced Micro Devices, www.amd.com, pp. 305-320, Sep 2003.

[18] C. Constantinescu, " Neutron SER characterization of microprocessors", *Proc. International Conference on Dependable Systems and Networks*, 2005.

[19] J.F.Ziegler et.al, "IBM experiments in soft fails in computer electronics", *IBM Journal of Research and Development,* Vol. 40, No. 1, 1996.

[20] T.Shanley et. al. "The Unabridged Pentium 4", *Addison Wesley Publishers*, 2004.