Pin Specific ESD Soft Failure Characterization Using a Fully Automated Set-up

Giorgi Maghlakelidze (1), Pengyu Wei (1), Wei Huang (2),
Harald Gossner (3), David Pommerenke (1)

(1) EMC Laboratory, Missouri S&T; 4000 Enterprise Dr, Rolla, MO, USA, 65401
tel.: 573-647-1573, e-mail: gmp73@mst.edu

(2) ESDEMC Technology; 4000 Enterprise Dr #103, Rolla, MO, USA
e-mail: whuang@esdemc.com

(3) Intel Germany GmbH, Am Campeon, Munich, Germany
e-mail: harald@intel.com

Abstract - A fully automated system is developed for the systematic characterization of soft failure robustness for a DUT. The methodology is founded on software-based detection methods and applied to a USB3 interface. The approach is extendable to other interfaces and measurement-based failure detection methods.

I. Introduction

In order to mitigate ESD-induced soft failures (SF) of a system, its robustness must first be evaluated. In light of the various parameters that influence the response of the system, it is best to use an automated characterization process. The outcome helps system-level and IC designers, and firmware developers.

The world of soft failures is diverse and has been studied in [1]-[8]. In this work, the device under test (DUT) is an Intel Joule 570x Internet of Things (IoT) platform. The USB3.0 interface was selected for characterization. USB3 related SFs were studied in [7]. Several disturbance methods were evaluated: system-level IEC, magnetic loop probe, conductive TLP injection, and directional injection. Soft failures were correlated to the stress parameters: the pulse rise time didn't seem to affect the failure threshold, while pulse width was found to be inverse proportional to it. The authors found no correlation between CPU stress and failure modes, but no other DUT load was explored. Furthermore, the root-cause analysis of more severe modes were performed and a strategy for SF-SEED was proposed. This work confirms some of the earlier findings and extends others. The main idea is to develop a software-based method for an automated and systematic pin-specific characterization, and to explore methods for such data processing that can extract useful information.

The automated system is able to provide quantitative information on the dependence of different failure thresholds on the injected pulse level, polarity, rise time, system load and state, pin-to-pin variation, etc. Eight failure types across four severity levels were identified for the given system and failure dependence on various system loads was established.

II. Characterization Process

The TLP injection system by ESDEMC [9] was used to deliver repeatable pulses to the DUT. The TLP system combined with an oscilloscope allowed the injected currents and consequential voltages to be measured. The TLP was controlled through GPIB and COM interfaces to the "Control PC", as shown in Figure 1.

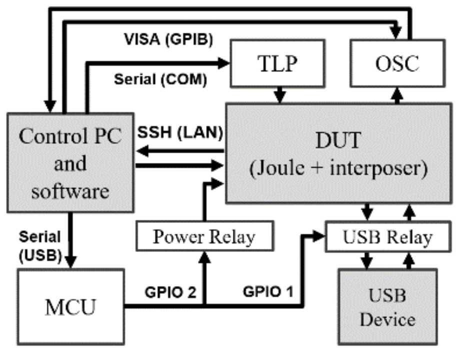

Figure 1: Overall system diagram.

Additional in-house software on the "Control PC" handled:

- The detection and recognition of failure modes,

- Sweeping of injected stress levels and polarities,
- Controlling the DUT over secure shell (SSH) protocol through the network (either through cable - LAN, or WiFi - WLAN), and
- Controlling other peripheral hardware (MCU).

A microcontroller unit (MCU), controlled over a serial interface, was used to switch two power relays: one for power cycling the DUT, another for tripping the power of the USB3 client device plugged into the host DUT port (interface under test).

A. Set-up description

1. Measurement set-up

The Intel Joule system consists of two separate parts – an expansion board and a compute module, as shown in Figure 2. The compute module contains all the key ICs (CPU, RAM, eMMC, Bluetooth, WiFi, etc.), while the expansion board provides power and fan-out to various interfaces (HDMI, microSD, USB3, USB-C, GPIO) with respective ESD protection devices. The compute module plugs into the expansion board through a 100-pin HiRose (HRS) surface-mount SF40 interconnect. In order to isolate the effects of the IC itself, rather than the effect of external ESD protection, an interposer board was developed. It was placed between the expansion board and the compute module, allowing injection of TLP pulses into the running (i.e., "hot") USB3 interface data lines of the DUT, without significant loading of the USB3 signals. This was achieved by using low-capacitance TVS diodes, an injection technique developed in [8]. The circuit is shown in Figure 3 and the board is shown in Figure 4.

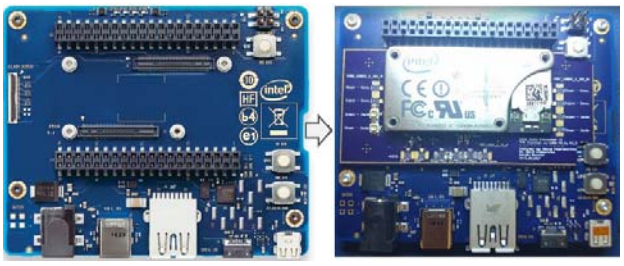

Figure 2: Left – expansion board with nothing pugged in; Right – compute module plugged into interposer, plugged into expansion board.

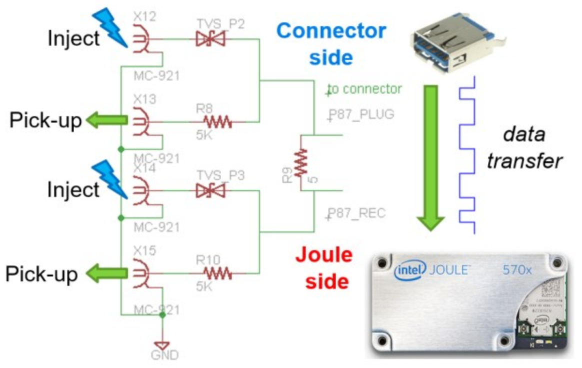

Figure 3: Injection and measurement circuit on the interposer board.

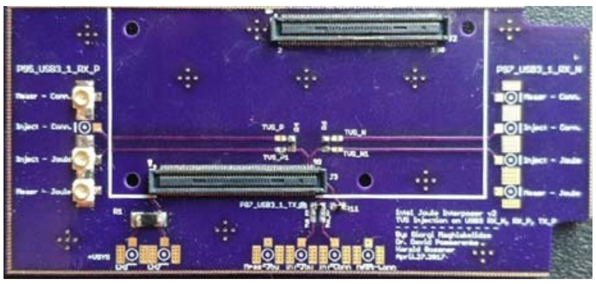

Figure 4: Populated interposer board photo, top view.

In the current work, three pulse lengths were used in the robustness evaluation: 100 ns, 6 ns, and 2 ns. The injection and measurement setup for the 100 ns pulse is presented in Figure 5. Figure 6 shows the setup for the 6 ns and 2 ns injections.

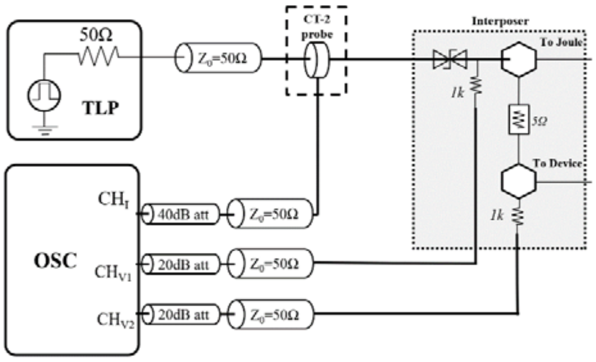

Figure 5: Injection and measurement setup diagram for 100 ns pulses (current deconvolution).

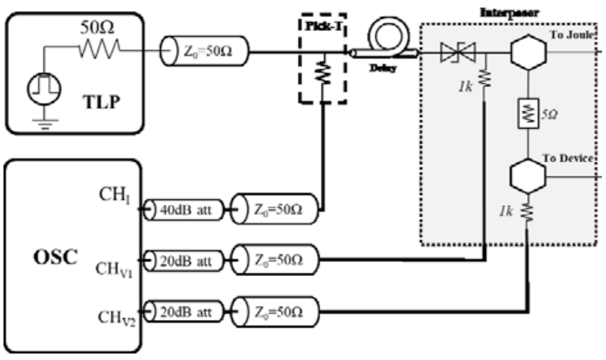

Figure 6: Injection and measurement setup diagram for 6 ns and 2ns pulses (vf-TLP method)

For the 100 ns injected pulse width, a current probe and deconvolution code was used to capture the injected current; for short pulses, a pick-off T combined with a delay line were used to separate the incident and reflected pulses (vf-TLP method).

The DUT and the peripheral hardware layout are depicted in Figure 7. The USB3 client device was a USB3.0 SanDisk memory stick. It is reasonable to expect that client-to-client variation will be minimal if TLP injection directionality is sufficiently high (i.e. the largest portion of the stress is injected towards the DUT, while the plugged in client experiences minimal stress).

2. Test Procedure for One Pulse

After calibrating the TLP injection and measurement system, the characterization procedure starts. For each injection level the following steps are taken:

1. Set the desired TLP voltage level;
2. Confirm that the DUT is in the "nominal" state (i.e. idle running and reporting);
3. Confirm that the interface under test is in the "nominal" state;
4. Inject a TLP pulse into the target pin;
5. Measure the waveforms and extract quasi-static voltage and current points;
6. Acquire kernel logs from the DUT;
7. Check if logs contain error messages;
8. Check if the interface under test is still in the "nominal" state;
9. If any abnormality is detected, classify and log the signature;
10. Detect soft failure mode;
11. Reset the USB interface to the nominal state (re-plug and check interface state);
12. If needed, reset the system to the nominal state;
13. Repeat for the next pulse level.

Each of the listed steps contains several sub-steps which complicate the characterization process. The full algorithm is discussed in the subsequent sections.

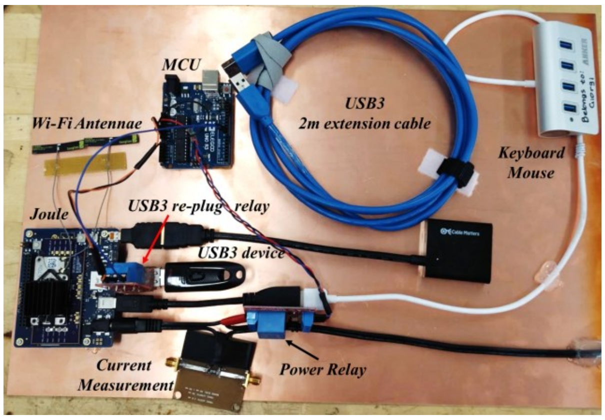

Figure 7: Photo of the DUT layout.

B. Automation Algorithm

The algorithm flow is almost fully depicted in Figure 8. The whole automated characterization process is run mostly from the "Control PC" by two separate software programs, along with an additional software program running on the DUT. One is the TLP software, and another is an in-house Python script. Voltage level, polarity, number of pulses, and number of injections for each level are set in the TLP graphical user interface. The TLP GUI also controls each injection and measurement, calibration, and current deconvolution. Upon a successful TLP injection, the GUI reports measured data to the Python script via an interface ASCII file, and proceeds to wait until the next injection is initiated. A "successful TLP injection" means that the current and voltage waveforms were measured without oscilloscope clipping and triggering problems. If clipping occurs, the TLP has to fire again in order for the scope to retrigger. This may cause system upsets without a proper V-I measurement. However, this happens only when the system transitions to a new stress injection level. Since each pulse is repeated

~100 times, sufficient information is collected to measure enough points for a quasi-static IV curve.

Upon receiving the data from the TLP software, the Python script pulls the kernel logs from the DUT via the SSH interface. The DUT runs Ubuntu GNU/Linux operating system, so by running the *dmesg* command [15] and filtering for USB-related events with *grep* command, the algorithm is able to establish whether a USB-related SF has occurred or not.

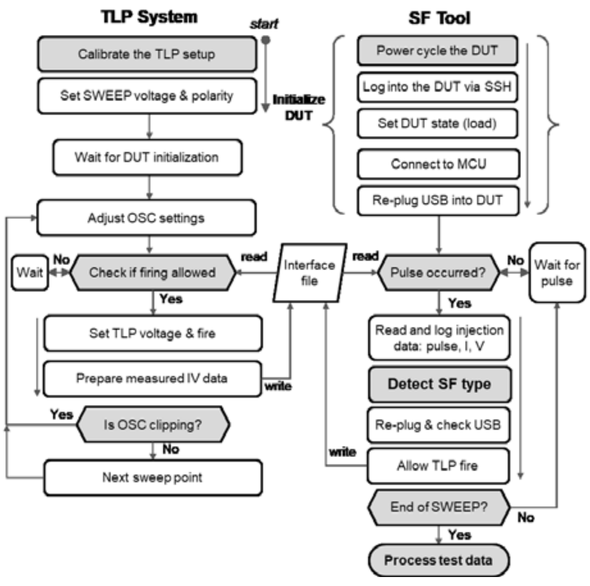

Figure 8: DUT SF characterization algorithm flow.

Some difficulty in the algorthm arises in three areas:

1. Bringing the DUT and the interface under test into the “nominal” state at every pulse;
2. Making sure that connections to the DUT and peripheral hardware are correctly opened and closed.
3. Differentiating between certain failure types based on the recovery method when the log message is unclear (e.g., failures that have similar signatures, but one requires a reboot, while the other – a power cycle).

These complications are caused by the SF and often manifest in the following ways:

- a disrupted connection to the DUT;
- causing a lost connection to the DUT due to a reboot;
- a need to reboot or power cycle the DUT to overcome the failure;
- a SF occurs, but no kernel message appears in the logs; the USB client device must be replugged to re-establish connection and re-evaluate the state of the DUT USB interface.

Obscurity of kernel messages can cause the algorithm to branch out and spend time “Detecting Soft Failure Type”, as depicted in Figure 9. The detection is rather simple: for each failure mode, there is a condition that needs to be satisfied. In the overall structure, there is a hierarchy of conditions that stack up from less severe to most severe. The left brach detects USB2 fallback-related failures, the right one detects USB3-related ones.

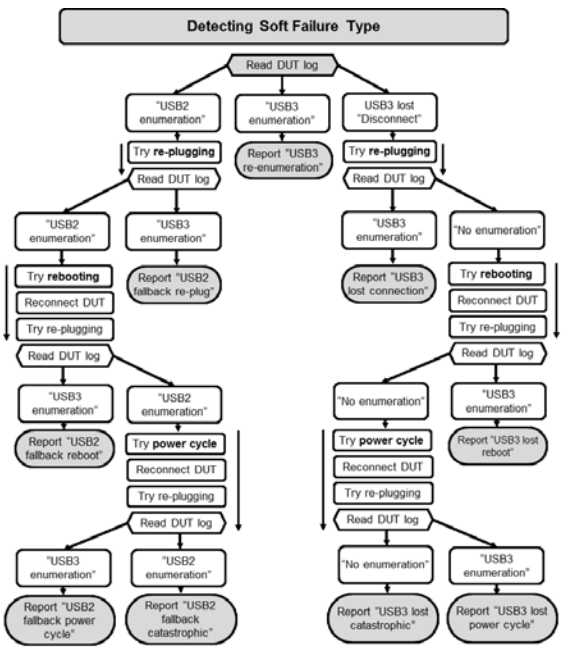

Figure 9: SF type detection algorithm.

Because SF behavior varies somewhat randomly, each test is performed up to 100 times. The data points (TLP voltage, injected current, voltage, polarity, state of the system, SF type) from each test are recorded in a *.csv file and later processed by a Python script using Pandas (code library used for big data analysis) [13]. The multi-dimensional data analysis is aided by constructing pivot charts grouped by the desired characteristic (e.g., injected current, pulse width, rise time, etc.) and calculating how often an SF type has occurred for each variation.

III. Results

A. ESD Gun testing

The Intel Joule development system was mounted inside an enclosure and a series of ESD gun tests were carried out. The purpose of the tests is to establish the range of soft failures when system-level stress is applied to different parts of the DUT: a) shield of USB3 port, and b) DUT chassis. An ESD Gun, Noiseken ESS-2000 TC-815R, was used to inject impulses in the range between 1 kV and 9kV, in contact discharge mode. The DUT and the injection points are shown in Figure 10. Each injection was repeated 100 times, while the operator monitored and logged occurring soft failures. Discharging into the DUT chassis (point 1 in Figure 10) was relatively robust, causing the HDMI screen to flicker several times at higher discharge voltages, but having no reported USB failures. Screen flicker is a kind of SF within the system, but unrelated to the USB3 interface, so it is not discussed in detail.

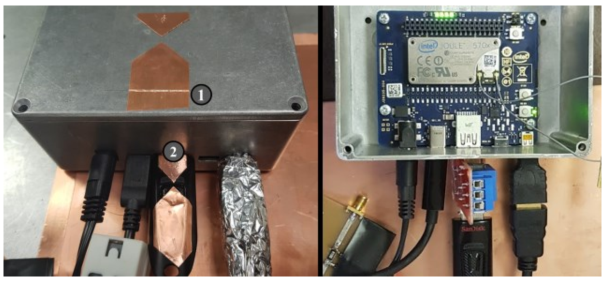

Figure 10: Left – injection points at the DUT chassis; Right – inside the chassis

The results of the ESD gun testing for system-level stress injected into the USB3 shield are shown in the Figure 11. Most of the soft failures are related to the HDMI screen (flickering, tinting with colors, screen turning off until HDMI cable replug). USB3 soft failures occur after 6kV, with a likelihood of <5% and varied severity: from re-enumeration of the device, to losing the connection and having to reboot the system. The logged failures correspond to the ones detected as a result of the automated characterization system, as discussed in detail in the subsequent sections.

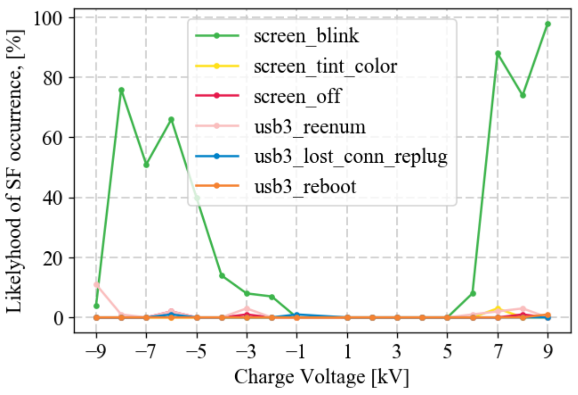

Figure 11: Results from ESD gun injection into the shield of USB3 interface of Joule expansion board

B. Soft Failure Classification

Observed soft failures can be categorized sufficiently well by Table 1 from [7], repeated here as Table 1. Category "A" is the least severe – the user does not notice the effect of failure and no intervention is required on their side. Category "B" is noticeable, but the system recovers without intervention (data transfer speed drops, the system reconnects to the client device, etc.). Category "C" is most severe and encompasses a varied family of failures, which may require as little as re-plugging the client device and as much as completely power cycling the DUT.

Table 1: Categories of Soft Failures as per [7]

Cat.	Definition	Example for USB
A	Operator does not notice, no operator intervention	Bit errors; packets getting resent
B	Operator notices, no operator intervention	Drop in data throughput; connection re-established by the host
C	Operator notices, intervention required	Stop of data transfer; re-plugging of the cable or power cycling required

The failure modes observed for the DUT mostly fall in the most severe category C. The full list and corresponding descriptions are in Table 2.

The most common SF is "USB3 re-enumeration" (Mode 2), which means that the DUT has re-established the connection with the client device without user intervention; in this case, USB3 functionality is preserved and no further action is required. Sometimes this failure mode is accompanied by a GUI error message which requires user interaction, making this variation a Category C failure. The next failure mode variation is "fallback to USB2" (Mode 3). It occurs as a result of negative current injection and requires user intervention. The milder case requires a mere re-plugging of the client device; a more serious case requires system reboot or power cycling. These take much longer than a re-plug: 60-90 seconds to reboot vs 5 seconds to re-plug, which may be a major inconvenience to the operator. In case of positive high-current injections, a rare failure occurs that disables the USB interface and requires re-plugging, rebooting or power cycling (Mode 4). Occasionally, Wi-Fi functionality is lost (Mode 5), but no correlation between injection level and its occurrence has been established.

The worst case for modern hand-held and wearable devices is the soft failure that requires physically disconnecting the power. For portable devices that would mean taking out and re-placing the battery or flipping a physical switch. Neither of these are an option for different design and policy reasons (waterproofing, warranty, security, etc.). This makes the requirements for such failures to be more stringent than less severe failure modes.

Table 2: Observed Failure Modes

Mode	Observation	Cat.
1	Drop in the data rate; no operator action required	B
2.1	Client device re-enumerated in USB3 mode; functionality restored by the system	B
2.2	Client device re-enumerated in USB3 mode, a GUI pop-up message occurs functionality restored by the system, but user has to click the message	C
3	Client device falls back to USB2 mode; 3.1 functionality restored by re-plugging the device 3.2 functionality restored by rebooting the DUT 3.3 functionality restored by power cycling	C
4	Client device disappears; 4.1 Functionality restored by re-plugging the device 4.2 Functionality restored by rebooting the DUT 4.3 Functionality restored by power cycling	C
5	Wi-Fi functionality is lost; functionality restored by power cycling the DUT	C

C. Variation of Pulse Length

The results for the Sandisk USB client for 2, 6, and 100 ns pulse width stresses are shown in Figures 12-14 respectively. The pulse levels are swept from -70 V to +170 V. The assymetry is explained by the high risk of hard failure if the negative stress is pushed to higher levels (at least for long-pulse case). The vertical axis is the likelihood of soft failure occurrence in percent; i.e., how often a particular SF has occurred out of all injected pulses for each particular pulse level and width.

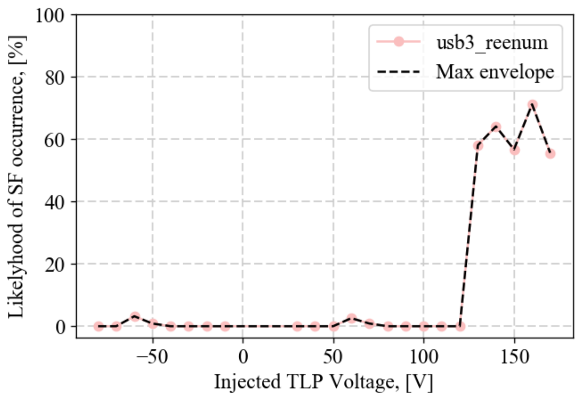

Figure 12: SF probability occurrence for 2 ns, against the TLP charge voltage.

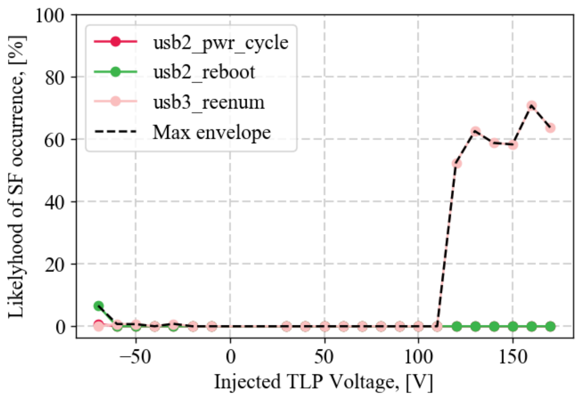

Figure 13: SF probability occurrence for 6 ns, against the TLP charge voltage.

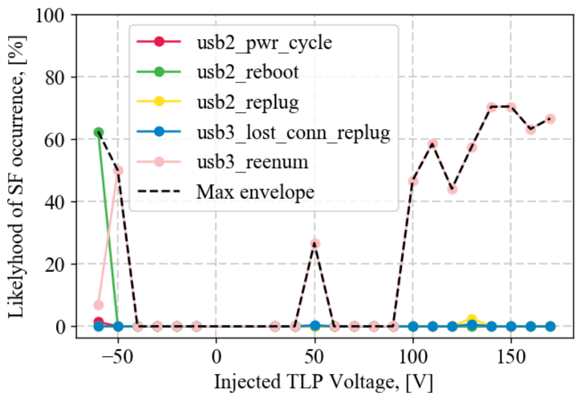

Figure 14: SF probability occurrence for 100 ns, against the TLP charge voltage.

The horizontal axis is the TLP charge voltage. As expected, at lower injection levels, no failures occur. For all cases, there seems to be a threshold, beyond which SF probability jumps from 0% to a substantial amount (between 50% and 80%). For lower duration pulses, this threshold is higher due to lower amount of energy delivered into the system. There seems to be little to no occurrence of serious soft failures for positive injections across the board. For positive current injections, only USB3 re-enumeration errors were observed. This is consistent across DUTs and other configurations. Only one case for the 100 ns injection had a somewhat severe fail – fallback to USB2, requiring the client to be re-plugged.

Negative current injections have a lower threshold and a richer variety of severe failure modes. USB enumeration failure rates are very small for short pulses, but increase to 53% from 0% at -50V TLP injection for 100 ns disturbance. However, the most interesting observation is that enumeration errors fall in frequency (to <10%) as other failure modes become prevalent – such as USB2 fallback requiring a reboot (~60%) or USB2 fallback requiring a power cycle (<5%), or USB3 connection loss, requiring a replug (also <5%).

This may indicate that some other sub-system is failing more severely than the one which leads to the USB enumeration failure. These tests were completed within several days and consist of over 15,000 data points. The results seem consistent with [8], in so far as exhibiting the inversely proportional relationship between the pulse width and the failure threshold. In this case, the novel information is that negative current injections cause far more severe failures and that shorter pulses seem to cause less varied and less severe failures for the same injection levels. The rise time dependence is not explored, as there is firm evidence [7] that the correlation is weak.

D. Variation of DUT System State

One of the parameters of interest is soft failure occurrence under different system load conditions. There is prior evidence that the CPU load doesn't have a significant influence on the likelihood of failure [7]. In this work, additional load conditions are explored by using a package *stress-ng* [14]. The package fully loads a 4-core CPU by using FFT function, reading and writing to RAM and eMMC. This load increases noise within the system, causing it to draw ~2x higher current and increasing overall system temperature. Hence, there is reasonable expectation that soft failures become more frequent, or more severe overall.

Ideally, one would repeat the full parametric sweep for each load condition. That increases characterization time many fold and is largely unnecessary, as baseline tests already show that no failures occur at lower injection levels. Therefore, in the interest of time conservation, only the threshold region for the positive injection sweep is selected for characterization under various load conditions. The results are shown in Figure 15.

The failure threshold stress current is the same for all cases and the occurrence levels vary between approximately 50% and 80%. Marginal variation from load to load is observed (within 10%). This confirms that the CPU load has only a weak influence on soft failure occurrence. RAM and eMMC loading shows similar results.

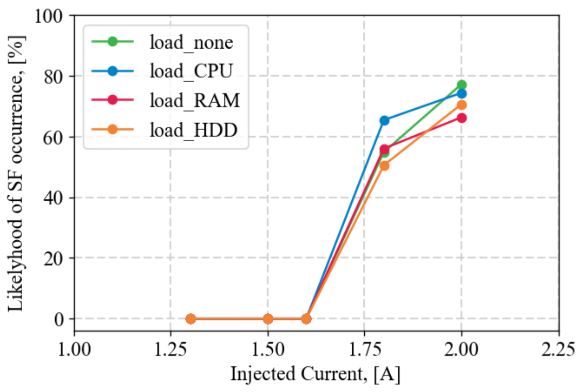

Figure 15: SF occurrence threshold due to positive injections under various system load states; 6ns injected stress.

It must be noted that the DUT load condition sweep was not automated in this case, but automation is possible with reasonably small effort. For this, during the stage "Set DUT State" in Figure 8, the operator defines several Linux command-line interface commands to be swept (one for each test condition) and one overarching loop is added that re-runs the algorithm for different load conditions.

IV. Discussion

The scope of this work is in automating the characterization flow and in expanding the knowledge about soft failure occurrence in complex systems. The root cause of specific soft failures is still being actively researched [3-6]. Specifically, with USB3 [7] [8] it has been found that more severe failures (fallback to USB2, etc.) occur due to power domain disturbances, while errors in data transmission are overwhelmingly consistent with lower-level pulses, where stress waveforms increase the signal peak-to-peak voltage.

One can draw practical conclusions from the obtained characterization data. From the expected ESD levels and the coupling paths, the designer can estimate the safe current waveforms and levels. These can be compared to the failure probability data from the IC characterization.

The system developer may establish a probability threshold for each failure mode and use the method for a "pass/fail" evaluation. Depending on the product purpose, 20% failure rate may be acceptable for SF not requiring operator intervention, while <1% may be acceptable for a SF that requires major actions like physical re-plugging or power cycling.

If soft failures are grouped, an envelope may be used to check the satisfaction of the passing criteria, e.g. "Max Envelope" on Figures 12-14.

A drawback of continuous, extensive TLP testing (especially with longer pulses) is the risk of "wearing out" the interface under test. This means introducing latent hardware failures by applying numerous pulses that under normal circumstances would not cause physical harm to the DUT.

Once the DUT is well characterized, the system designer can use that information to "get it right the first time" and/or reduce the number of product development iterations:

1. Make system design changes to mitigate some SF (system-, circuit-, and IC-level). This is especially beneficial in the early design stages of a product, when a designer is able to introduce additional protection, filtering, shielding, etc.
2. Make firmware or software improvements that would reduce severity or frequency of specific failure modes.

In cases that require inclusion of a measurement-based method (e.g. spike in current consumption of the interface) [4] [11] [12], at first it should be tested independently to establish the reliability and efficacy of the measurement method. Once the clear detection criteria are established, a function within "Detect SF Type" in Figure 9 can check if the criterion for detecting the SF has been satisfied.

In order to adapt this characterization method to a different interface, at first exploratory work must be done to establish the variety of soft failure modes. Then hardware and software efforts are carried out. In terms of hardware – auxiliary boards may need to be designed to facilitate re-plugging, power cycling the interface of interest, etc. In terms of software – a function set within "Detect SF Type" must be written. These functions inquire and establish whether the criteria for SF detection have been met. In addition, interface initialization functions may require change. The rest of the algorithm largely remains the same.

V. Conclusion

An automated system for SF robustness characterization was developed and applied to a USB3 interface of an existing development platform for a number of stress pulse lengths and system load conditions. Test results were processed and soft failure occurrence likelihood statistics were obtained for various levels of TLP injections, and both polarities. In the scope of this work, software-based detection methods were utilized, but the methodology is extendable to other interfaces and measurement-based failure detection methods as well.

The methodology has a wide application range, but is possibly most useful for high-reliability systems that could not tolerate soft failures. One of the directions for further research is a deeper investigation into SF occurrence depending on system states (CPU load, GPU load, etc.) and a wider range of disturbances. Characterization and data processing methods are well established and may be extended for further study.

Acknowledgements

The material is based upon work supported by the National Science Foundation, Grant IIP-1440110.

The authors would like to thank Nicholas Erickson of Missouri S&T EMC Laboratory for constructive criticism of the manuscript.

References

[1] Industry Council on ESD Targets, White Paper 3: System Level ESD, Part II: “Implementation of Effective ESD Robust Designs”, September 2012

[2] Duvvury, C. and Gossner, H., “System level ESD co-design”, 1st ed. Wiley - IEEE, 2016

[3] N. A. Thomson et al, "Soft-Failures Induced by System-Level ESD," in IEEE Transactions on Device and Materials Reliability, vol. 17, no. 1, pp. 90-98

[4] S. Yang *et al*, "Measurement techniques to predict the soft failure susceptibility of an IC without the aid of a complete software stack," IEEE EMCS 2016, pp.41-45

[5] B. Orr et al, "A systematic method for determining soft-failure robustness of a subsystem," 2013 35th EOS/ESD Symposium, Las Vegas, NV, 2013, pp. 1-8.

[6] S. Vora, R. Jiang, S. Vasudevan and E. Rosenbaum, "Application level investigation of system-level ESD-induced soft failures," 2016 38th Electrical Overstress/Electrostatic Discharge Symposium (EOS/ESD), Garden Grove, CA, 2016

[7] S. Koch et al “Identification of Soft Failure Mechanisms Triggered by ESD Stress on a Powered USB 3.0 Interface”, accepted by IEEE Transactions on EMC, 2018

[8] G. Notermans, H. M. Ritter, B. Laue and S. Seider, "Gun tests of a USB3 host controller board," 2016 38th Electrical Overstress/Electrostatic Discharge Symposium (EOS/ESD), Garden Grove, CA, 2016

[9] ESDEMC Technology: https://www.esdemc.com/

[10] T. Schwingshackl et al., "Powered system-level conductive TLP probing method for ESD/EMI hard fail and soft fail threshold evaluation," 2013 35th EOS/ESD Symposium, 2013, pp. 1-8.

[11] J. Zhou, Y. Guo, S. Shinde, A. Patnaik, , O. H. Izadi, C. Zeng, , J. Shi, J. Maeshima, H. Shumiya, K. Araki, and D. Pommerenke, “Measurement Techniques to Identify Soft Failure Sensitivity to ESD” accepted by IEEE Transactions on EMC, 2018

[12] O. H. Izadi, A. Hosseinbeig, H. Shumiya, J. Maeshima, K. Araki, D. Pommerenke, “Systematic Analysis of ESD-Induced Soft-Failures As A Function of Operating Conditions”, 2018 Asia-Pacific International Symposium on Electromagnetic Compatibility (APEMC), Singapore, 2018

[13] Python Data Analysis Library *pandas*: https://pandas.pydata.org/

[14] Stress-testing package *stress-ng* for GNU/Linux: http://manpages.ubuntu.com/manpages/artful/man1/stress-ng.1.html

[15] Driver message command *dmesg* for GNU/Linux: http://man7.org/linux/man-pages/man1/dmesg.1.html