Journal of Circuits, Systems, and Computers
Vol. 28, Suppl. 1 (2019) 1940003 (15 pages)

DOI: 10.1142/S0218126619400036

A Digital On-Line Monitor for Detecting Intermittent Resistance Faults at Board Level*

Hassan Ebrahimi† and Hans G. Kerkhoff‡

Testable Design and Test of Integrated Systems (TDT) Group,
Centre for Telematics and Information Technology (CTIT),
University of Twente, Enschede, The Netherlands
†*h.ebrahimi@utwente.nl*
‡*h.g.kerkhoff@utwente.nl*

Received 9 October 2018
Accepted 25 March 2019
Published 15 May 2019

The reliability of board-level data communications intensively depends on the reliability of interconnections on a board. One of the most challenging interconnections reliability threats is intermittent resistive faults (IRFs). Detecting such faults is a major challenge. The main reason is the random behavior of these faults. They may occur randomly in time, duration and amplitude. The occurrence rate can vary from a few nanoseconds to months. This paper investigates IRF detection at the board level by introducing a new digital *in situ* IRF monitor. Hardware-based fault injection has been used to validate the proposed IRF monitor. As case studies, two widely used on-board transmission protocols namely the Universal Asynchronous Receiver Transmitter (UART) and the Serial Peripheral Interface bus (SPI), have been used. In addition, one fault management framework, based on the IJTAG standard, has been implemented to collect and characterize information from the monitors. The experimental results show that the proposed monitor is effective in detecting IRFs at the board level.

Keywords: Reliability; no faults found; intermittent resistive faults; intermittent fault detection; board-level fault detection.

1. Introduction

The complexity of circuit boards continues to grow rapidly. The number of devices on a board such as sensors, actuators and memories is increasing. It leads to having more data communications at board level. The reliability of on-board data communications highly depends on the reliability of interconnections. One of the greatest interconnect reliability challenges that threats the dependability of highly dependable systems are intermittent resistive faults (IRFs).

*This paper was recommended by Regional Editor Zoran Stamenkovic.
†Corresponding author.

IRFs are a specific category of No Fault Founds (NFFs). Like other types of NFFs, they result in many product returns in the automotive and avionics industries.[1] The most common cause of IRFs is marginal or unstable interconnections. There are varieties of interconnections in circuit boards which are vulnerable to IRFs, such as tracks, vias and solder joints. In addition, the increased instability in interconnections caused by electromigration, corrosion, temperature and mechanical stress can trigger IRFs occurrences.

IRFs might randomly occur in any interconnection of a system during its operational mode. They manifest themselves as a sequence of small resistance changes in an interconnection. They may emerge repetitively in a location and may gradually become increasingly severe during the lifetime of the system. Finally after a while, they may evolve into a permanent fault.[2] Therefore, it is vital to detect and repair such faults before they become permanent and result in a system failure especially in safety-critical systems.

Intermittent fault detection is difficult. The main reason is intermittent faults are not deterministic and may not appear during testing. Therefore, traditional testing methods are not effective for intermittent fault detection. One approach toward intermittent fault detection is using periodic testing.[3] In periodic testing, the probability of detecting intermittent faults increases by retesting the system periodically. Another approach is continuous monitoring of the behavior of a circuit using various embedded instruments during the operational mode.[4]

On-line monitoring can be used at both the board[4] and the chip[5] levels for IRF detection. In Ref. 4, we presented an extended version of the IEEE standard 1149.4 which gives the ability of on-line monitoring of wiring tracks in boards. In this paper, a more cost-effective approach by using fully digital *in situ* on-line monitors is investigated. For this purpose, a new fully digital on-line monitor to detect IRFs in data communications at the board level is introduced. In addition, a fault-management framework based on the IJTAG standard has been developed to facilitate IRF detection and localization. Finally, the efficiency and capability of the new monitor have been validated by hardware implementations and fault injection.

The rest of the paper is organized as follows. Section 2 reviews related work and the background to fault detection at the board level. The proposed monitor is introduced in Sec. 3. Section 4 introduces the fault-management framework. Section 5 shows IRF detection at the board level. First, an IRF model and the fault-injection framework are introduced. Then, the experimental setup and the measurement results of fault-injection experiments are presented. Finally, conclusions are drawn in Sec. 6.

2. Related Work

Several methods have been used traditionally for testing interconnects at the board level such as DC resistance measurements,[6] radio-frequency (RF) impedance analysis[7,8]

Built-in Self-Test (BIST)[9] and analog neural network technology.[10] DC resistance measurements[6] is a useful method to detect stuck-at short, stuck-at open and bridge faults in interconnections at test mode. However, for detecting small impedance changes in interconnections, RF analysis is more effective.[7]

RF analysis has been widely used for detecting discontinuities in wire and cable connections. This measurement technique can be used in both the time as well as frequency domain. Time-domain reflectometry is based on transmitting an RF signal through a wire and observing the reflections. If there is a discontinuity in the wire, an impedance mismatch will be observed. Another reflectometry technique is frequency-domain reflectometry (FDR). FDR can detect a fault location in a transmission line by measuring frequency, amplitude and phase changes. FDR has been utilized in Ref. 8 to monitor degradation in connectors.

Another technique to test interconnection reliability at the board level is based on BIST. A method based on BIST, referred to as SJ BIT, was proposed in Ref. 9. It can detect faults induced by solder-joint fractures in the input/output pins of field programmable gate array (FPGA) devices.

All of the techniques mentioned above are powerful methods for testing discontinuity or impedance degradations in interconnections and wires in test mode. Because of the nondeterministic behavior of intermittent faults, the probability that an IRF is activated while in test mode is very low. Therefore, the conventional test methods are highly unlikely to detect these faults. Furthermore, even if one is able to evoke an IRF during the test phase, very accurate equipment at high frequencies such as data logger,[6] vector network analyzer,[7,8] and analog neural network[10] are needed to detect brief IRFs. Therefore, since IRF may remain undetected during a test phase, an *in situ* on-line monitoring technique should be used to detect such faults when they become active while a system is in operational mode.

In situ on-line monitoring is widely used at the chip level for timing-fault detection.[11–13] Timing violation along a data path can occur due to process, voltage and temperature variations.[14] In addition, aging and IRF[15] can cause timing violations in the data path. *In situ* on-line monitoring techniques have been used for timing error detection and correction at the critical paths,[11] timing slack measurement,[13] adaptive voltage scaling[16] and aging detection.[17] With some modifications, the *in situ* on-line monitoring techniques can be used for IRF detection at the chip levels as well.[5]

For adapting on-line monitoring techniques for IRF detection, we need to focus on the difference between intermittent faults and other timing faults. The main difference is that IRFs can cause random timing violations. The IRF monitor should be able to detect this random behavior. Another difference is that they can occur randomly in any location. Therefore, a fault localization mechanism is required for IRFs detection. In addition, because IRFs may reoccur at the same location, a fault-management software is required to keep track of the occurrence rate and the behavior of faults. To fulfill these requirements, we have designed a fully digital *in situ* IRF monitor which is compatible with the IJTAG standard.[18] In addition, we have

implemented a fault-management framework with fault localization and classification capabilities.[19] In the following sections, the proposed monitor and the fault management framework are presented in detail.

3. The Proposed Intermittent Resistive Fault Monitor

An excellent health measurement of a data transmission is the continuous monitoring of its timing constraints to ensure there are not any timing violations.[12] Measuring the timing slack at the end point of the transmission line can determine the health of the communication. Briefly, a positive timing slack indicates that a data communication is healthy, with a margin by which the delay can increase before causing an error. A small slack is an indication that the communication is close to fail. Measuring timing slack can give an early warning of deterioration and triggering preemptive actions to avoid failure because of IRFs or other timing violations. Based on this concept, we have designed our new IRF monitor. The proposed IRF monitor continuously monitors signals at the end of the transmission line. It ensures there is never a timing violation during operation so as to avoid incorrect data being captured.

Figure 1 shows how the IRF monitor can be used at the end of a serial transmission line. It can be a part of the receiver or a separated module. To be able to detect IRFs, the monitor needs to receive three signals: the transmitted data at the end of the transmission line (*Data*), the data captured by the receiver (*Captured Data*) as well as the system *Clock* signal. The output of the monitor is a *Warning* signal which will be active if a timing violation is detected.

To make sure that a data communication will perform correctly, the data signal at the end of a transmission line should stay stable during a timing window (detection window); otherwise the receiver may capture wrong data. The IRF monitor checks whether or not a late transition in the transmission line occurs. The detection window can be generated using a guard-band signal[19] or delay elements. In this paper, a delay chain is used to create the detection window.

The schematic of the proposed IRF monitor is shown in Fig. 2. It consists of two D-flip-flops, a delay chain and a comparator. The flip-flop colored in red is a part of the receiver and not the monitor. The comparator checks if the same data is captured by the monitor and receiver. In the case of discrepancy, a warning signal will be generated. The data can be captured at the rising clock edges. At the clock falling edges, the captured data in the receiver and the monitor will be compared. If a

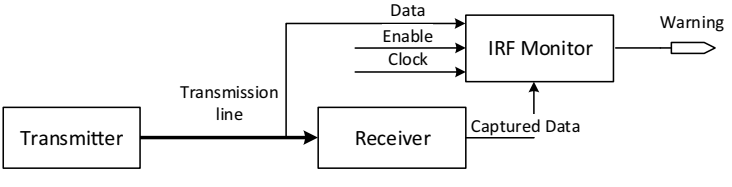

Fig. 1. An example of a transmission line equipped with an IRF monitor.

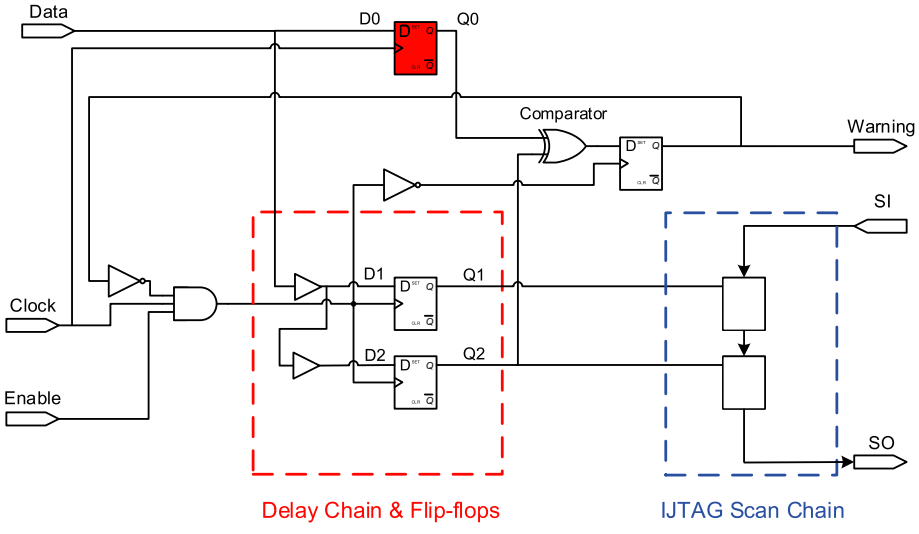

Fig. 2. The schematic of the proposed IRF monitor.

warning occurs, the degree of timing violation that is captured in the flip-flops will be transferred to the IJTAG chain. It should be noted that the number of the flip-flops and the length of the delay chain can vary based on the required precision. Here, two flip-flops are sufficient for the fault-management software to distinguish the severity of violations and classify them into transient, intermittent and permanent faults. The clock signal in the monitor is gated by the *Enable* and *Warning* signals. It means that if either the monitor is not enabled or a warning is detected, then the monitor will not capture a new timing violation and it should be enabled or reset via the IJTAG network.

An example of an IRF detection is shown in Fig. 3. In this figure, $D0$ is the data at the end of the transmission line. $D1$ and $D2$ are the data after passing through the delay chain. $Q0$ is the data captured at the rising clock edge by the receiver. $Q1$ and $Q2$ are the data captured in the flip-flops ($Q2 = 1$) of the monitor. The Figure shows

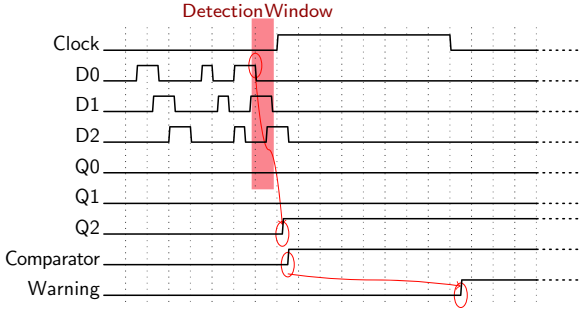

Fig. 3. An example of an IRF detection.

how the monitor detects a late transition in the transmission line. As can be seen, the $Q0$ signal has a high-to-low transition in the detection window, as a result a timing violation is captured in the monitor's flip-flops ($Q2 = 1$). The *Comparator* signal shows the comparison of the $Q0$ and $Q2$ signals and since they are not equal, the *Warning* signal is raised at the falling clock edge.

4. Fault Management Framework

The stored data in the flip-flops of the monitor will be read by a fault-management framework via an IJTAG[18] network. Based on the degree of a violation, the fault location and its occurrence rate, the fault-management software can classify the type of the detected fault: whether it is caused by a transient, intermittent or permanent fault. In short, if during a certain period, several faults with various degrees of timing violation occur in a line, it can be classified as an IRF. If only one timing violation occurs, it can be classified as a transient fault. Finally, it would be considered a permanent fault if the number of violation occurrences is higher than a certain threshold.

Figure 4 shows the IRF monitor wrapped in an IJTAG wrapper. The IRF monitor is the same as that shown in Fig. 2. A simple access procedure written in PDL for the

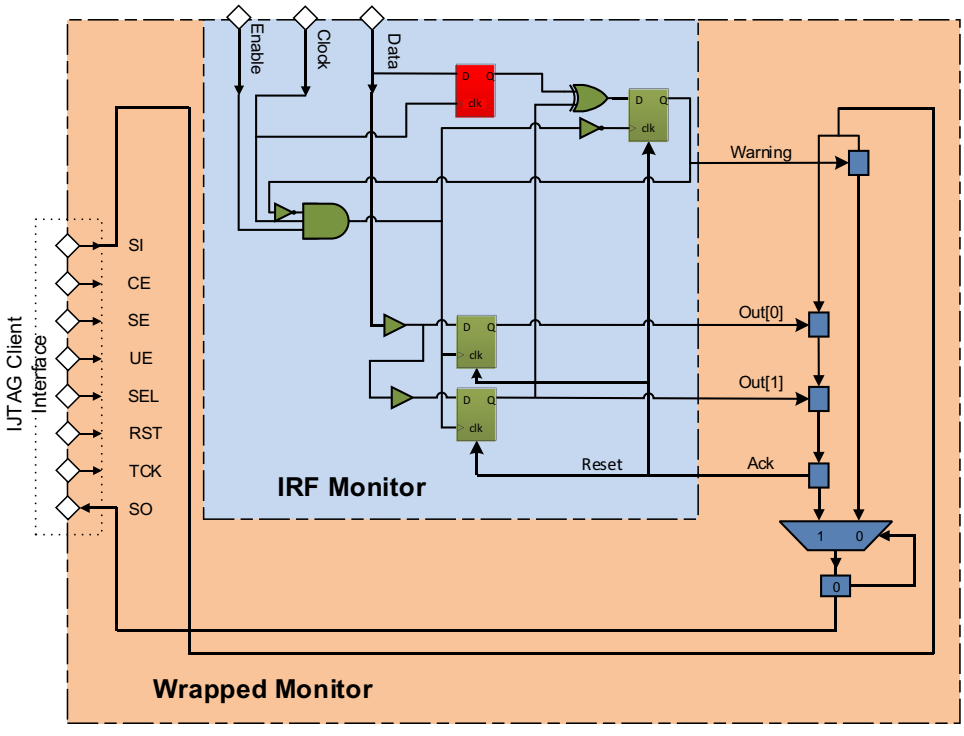

Fig. 4. An IRF monitor wrapped by the IJTAG standard.

```
1  iPDLLevel 1
2  iProcsForModule IRF_Monitor
3  set Status 0
4  while {$Status == 0} {
5      iRead IRF_Monitor.Warning
6      iApply
7      set Status [iGetReadData IRF_Monitor.Warning]
8  }
9  iRead IRF_Monitor.Out
10 iWrite IRF_Monitor.Ack 0b1
11 iApply
12 set IRF_Out [iGetReadData IRF_Monitor.Out]
13 iWrite IRF_Monitor.Ack 0b0
14 iApply
```

Fig. 5. Simple example of a PDL file of the IRF monitor wrapped by the IJTAG network.

IRF monitor is presented in Fig. 5. It consists of a simple logical synchronization mechanism between the IJTAG controller and the monitor. This controller continuously polls the warning flag until a warning is raised (lines 4–8); then it reads the output data and concurrently writes a *1* in the acknowledge register in order to instruct the instrument to reset its warning and data outputs (lines 9–11). Finally, the controller resets the acknowledge flag in the next scan cycle (lines 13–14) to allow the IRF monitor to restart the fault monitoring process.

The fault management consists of software and hardware parts. The fault management software comprises a Matlab script plus a JTAG Live software[20] script. The Matlab script stores and analyzes the information collected from the monitors. Whereas, the JTAG Live script is responsible for configuring and collecting data from the monitors through the IJTAG network. The structure of the IJTAG network and the location of monitors can be provided to the JTAG Live script by a file written in the ICL language format. Reading and writing procedures from/to the monitors can be specified by the PDL language. It should be noted that the ICL and PDL languages are parts of the IJTAG standard.[18]

5. IRF Detection at Board-Level

In this section, the effectiveness of the IRF monitor with regard to board-level IRF detection will be investigated. First, the IRF model which has been used in the fault-injection experiments is introduced. Then, the fault-injection framework is described. Finally, experimental results are presented.

5.1. *Intermittent resistive faults model*

The IRFs have been modeled based on some IRF measurements.[2,5] In this model, a burst of IRF pulses is generated based on six parameters. These parameters are the

Table 1. Range of used parameters in the IRF generator.

Parameter	Minimum	Maximum	Distribution
Burst length	1	5	Uniform
Resistance	1.0 Ω	2.5 kΩ	Uniform
Start time	0.1 μs	2.0 μs	Uniform
Active time	0.2 μs	1.5 μs	Uniform
Inactive time	0.2 μs	1.0 μs	Uniform
Safe time	1.0 μs	20 μs	Uniform

following: The number of pulses in a burst (burst length). The start time of the burst. A resistance range of the IRF where the resistance value of each pulse will be chosen randomly. Finally, the active and inactive times range. After determination of the range of these parameters by the user, the IRF generator software generates random resistance pulses according to these parameters and sends IRF information to the hardware fault injector.

Table 1 lists the values and distributions that have been used for the fault injection in this paper. After a random *Start Time* (in range of 0.1–2 μs) from the beginning of fault injection, the burst of resistance pulses starts. Each pulse in the burst has a random resistance value that is active for a random activation time (*Active Time*). There is an inactivation time (*Inactive Time*) between two pulses. During this time, there is a fault-free situation ($R = 1\Omega$). At the end of an IRF, the IRF generation procedure is completed by generating a fault-free situation (safe time).

5.2. *Fault injection framework*

The fault-injection framework is composed of fault-generator software and fault injector hardware. The fault-generator software running on a personal computer is responsible to generate IRFs based on the IRF model. The generated IRF can be transferred to the fault injection hardware via a serial communication link. The fault-injection hardware[21] is composed of a custom printed circuit board and an FPGA board (see Fig. 7). The FPGA accurately controls some fast digital switches and potentiometer on the printed circuit board to generate the sequences of IRFs. Figure 6 shows how the fault management and the injection frameworks have been used for the fault injection and measurement in this paper.

5.3. *Experimental setup*

To evaluate our new IRF detection technique, two widely used serial protocols for on-board communication, namely the Universal Asynchronous Receiver Transmitter (UART)[22] and the Serial Peripheral Interface bus (SPI),[23] were used as case studies. Transmitters and receivers for both protocols were designed at the logic gate level and implemented on an FPGA. The experimental setup of the fault injection and

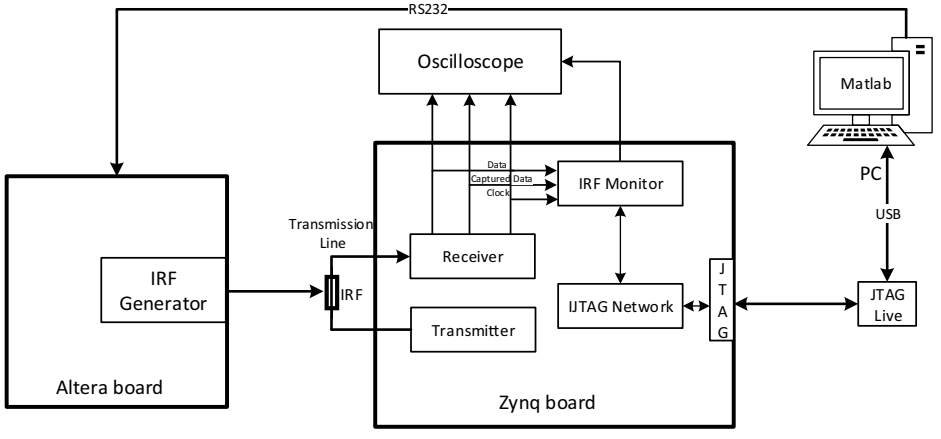

Fig. 6. A block diagram of the usage of the fault management and injection frameworks in the IRF injection and measurement set-up.

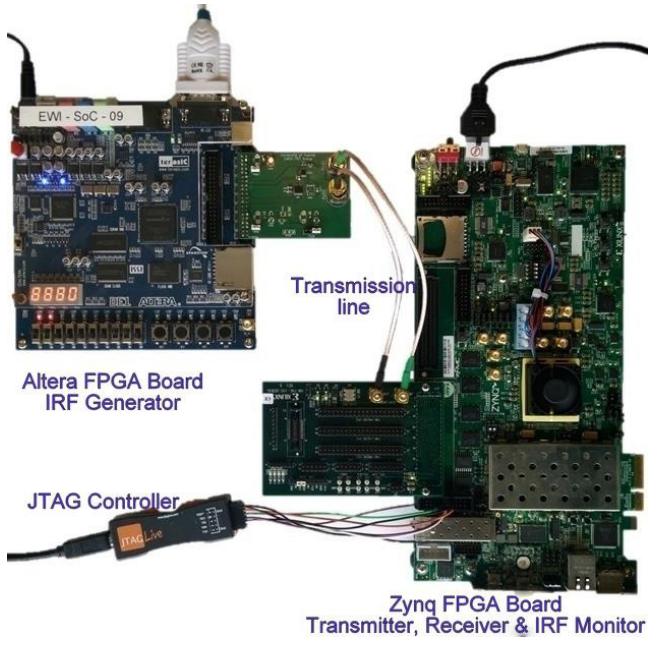

Fig. 7. Experimental set-up of the IRF injection and monitoring.

monitoring is shown in Fig. 7. In this setup, two FPGA evaluation boards were used: an Altera DE1 board for the fault generator and a Zynq-7000 board for implementing the transmitter and the receiver as well as the IRF monitor and the IJTAG network. It should be mentioned that the transmitters could be implemented on a separated board but for simplicity they were implemented on the same board with the

receivers. As can be seen, the fault generator was connected to the transmission line in the middle. In practice, the IRF generator is connected in series with the transmission line. By this construction, we could emulate a defected transmission line. In our experiments, the transmission baud rate and power supply were set to 3 MHz and 2.5 V, respectively.

During transmission, IRFs were injected in the transmission line by the fault-injection framework and simultaneously monitored by the fault-injection framework. The parameters which were used by the fault-injection framework to generate the injected IRF are shown in Table 1. The injected pulses are randomly generated with resistances ranging between 1 Ω and 2.5 $k\Omega$. This range is relatively small and hence less likely to cause logical errors in the data transmission line, but still can cause timing distortions. Therefore, it is suitable for evaluating the monitor.

5.4. *Measurement results*

An example of IRF injection and measurement in a UART transmission is shown in Fig. 8. The voltage scale is 2 V and the timescale is 0.67 μs. Conventionally, in a UART transmission every frame of data starts with a start bit, continues with bits of data and ends with one or none parity bit as well as one or two stop bits. In our experiment, the data is 8 bits that is transferred from the less significant bits to the most significant bit of the data. An even parity is used and the number of stop bits is two.

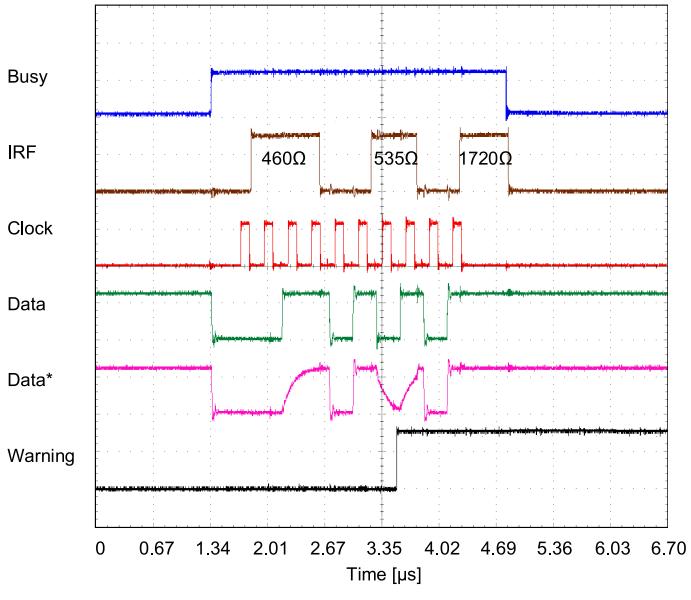

Fig. 8. Measured waveform results of the proposed IRF monitor under hardware-based IRF injection for the UART protocol.

In Fig. 8, the *Busy* signal shows the moment when the receiver is busy with receiving data. *Data* and *Data** show the bit stream received by the receiver before and after fault injection in the transmission line, respectively. The *IRF* indicates which amplitude and duration and when the IRF has been injected. The *Clock* signal shows when the data has been captured by the IRF monitor and the UART receiver.

The injected IRF is a burst consisting of three pulses with the values of 460, 535 and 1720 Ω, respectively. The first pulse has occurred during 1.8 and 2.6 μs. This pulse has distorted the Data signal, and hence the rising time at the Data* signal has increased during this period. Although the injected IRF at this duration has caused a timing violation, its amplitude was not sufficient to be detected by the monitor. The second pulse was active between 3.20 μs and 3.75 μs. It has induced a large timing violation and is detected by the IRF monitor, and as a consequence the Warning signal has become a logic one. The last pulse had a relatively large resistance value, but it has not occurred during a signal transition, it had no effect on the Data signal. It should be noted that the IRF has not caused a logic error and the sent bit stream *10101100* has been transmitted intact without parity or frame error. However, the IRF monitor detected the timing violation caused by the injected IRF. This shows the ability of the monitor in IRF detection in early stages before IRFs will cause an undetectable data error.

Figure 9 shows one example of IRF injection in a transmission line with the SPI protocol. A master and a slave SPI have been implemented. The clock polarity is

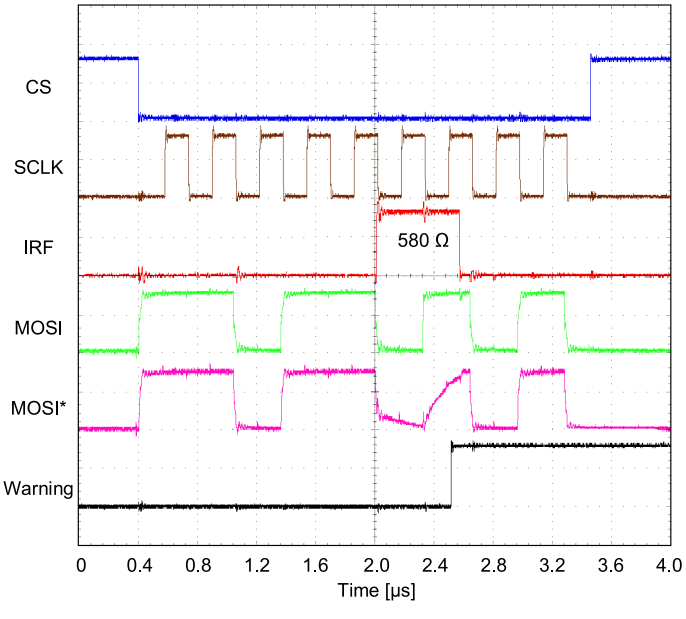

Fig. 9. Measured waveform results of the proposed IRF monitor under hardware-based IRF injection for the SPI protocol.

noninverted. It means the clock is idle at 0, and the first clock edge is a rising edge. The clock phase is CPHA = 0. It means data is captured at the leading edge of the clock.[23] Each frame of data starts with the less significant bit of eight bits data and ends with an even parity. In Fig. 9, the signals *CS*, *SCLK* and *MOSI* are chip select, SPI clock and Master Output Slave Input, respectively. The *MOSI** signal shows the *MOSI* signal after IRF injection. The *IRF* signal indicates the injected IRF by the IRF generator in the MOSI line. The injected fault has one pulse with the resistance value of 580 Ω and a duration of 0.56 μs. The fault is detected by the IRF monitor at time 2.45 μs, and as a result the Warning signal has become a logic one.

Generally, serial transmissions can be protected against some logic errors by framing check or adding some parity bits. We have investigated vulnerability of two widely used error-checking mechanisms against IRFs: being single parity and Hamming (7,4) coding. A single parity mechanism can detect an error in the case of an odd number of bits are being flipped during a transmission. A Hamming (7,4) error check mechanism is similar to the single-parity mechanism, but it utilizes three parity bits for four data bits instead of one single parity for a byte of data. In addition, a framing check was implemented by checking the number of received bits in each frame of data. If the number of received bits of a frame is not matched with the default frame size, a frame error occurs.

For each protocol, a total of 20,000 fault injections have been performed on random data transmissions. Figure 10 shows the result for the UART protocol transmission for two cases of single parity and Hamming (7,4) error-checking mechanisms. It should be noted that the result for the UART and SPI protocols are similar, but for the sake of brevity only the result for the UART protocol has been

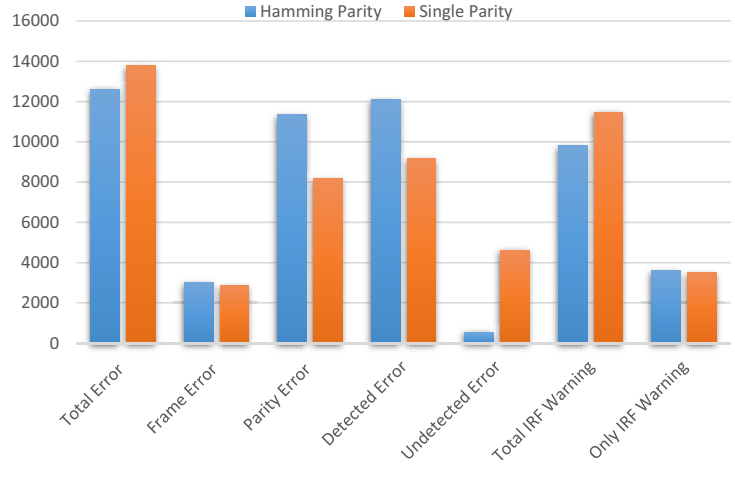

Fig. 10. Fault injection results for cases of Hamming and single parities error detection.

shown here. In Fig. 10, *Total errors* shows the total amount of logic errors. A logic error happens in the case the transmitted data is not the same as the received data. This number has been calculated on the top level by the fault-management framework. The total number of logical errors are about 62% and 68% of the injected faults in the cases of the Hamming and the single-parity mechanisms, respectively. In the case of the Hamming parity mechanism, 95% of the total logical errors are detected by either parity or frame error check or both. This ratio is 66% for the single-parity mechanism. As expected, the Hamming mechanism can detect more logic errors than the single-parity mechanism. However, because of the limitation of the parity mechanisms some logic errors still remain undetected. The ratio of undetected logic errors is 5% and 34% for Hamming and the single-parity mechanism, respectively.

As was mentioned earlier, even with parity and frame checking, some logic errors may still remain undetected during a serial data transmission. Using the IRF monitor in addition to the error-detection mechanisms gives an extra ability to the serial receivers to detect IRFs at early stages before they lead to undetected logic errors. In Fig. 10, the value of "*Total IRFM Warning*" indicates that the proposed monitor could detect a timing violation in more than 50% of the total number of the IRF injections. The value of "*Only IRFM Warning*" is important. It shows how many times the injected IRFs have caused timing violations in the transmission line, but yet without causing any logical (parity or frame) errors. This happens at early stages of IRFs. Almost 25% of our IRF injections consisted of early stages (with resistances below 1.2 KΩ). The proposed monitor was able to detect all the injected IRFs with resistance values higher than 0.46 KΩ and duration longer than 0.1 μs. Those IRFs that have resistance values exceeding 1.2 KΩ caused either parity or frame or both errors.

To have an accurate estimation of the area and power overhead of our design, we have implemented all modules at the logic gate level and synthesized them using TSMC 40 nm technology. The results of area and power consumption per module are listed in Table 2. The area scale is in square micrometers and the power consumption is in micro Watts. The dynamic power consumption has been calculated based on a clock of 50 MHz frequency. As can be seen, the IRF monitor has a small area and

Table 2. Area and power consumption overheads for all modules.

Modules	Area [μm^2]	Power [μW]
UART RX	606.3	15.9
UART TX	512.0	15.1
SPI Slave	418.6	1.7
SPI Master	805.0	11.6
IRF Monitor	40.2	0.9
IJTAG Network	329.8	6.4
TAP Controller	1003.1	6.1

power consumption overhead. For UART and SPI protocols, the area overheads are 3.6% and 3.3%, and the power consumption overheads are 2.9% and 6.8%, respectively.

6. Conclusions

A fully digital on-line monitor has been presented to address one of most challenging interconnection reliability problems being IRF at the board level. The new proposed monitor has been evaluated in actual hardware using our fault-injection and fault-management frameworks. The experimental results show that with a small area and power overhead the proposed monitor gives the ability to detect IRFs at the board level in their early stages and before they lead to logic errors.

Acknowledgments

This research was carried out within the Horizon 2020 IMMORTAL project, financed by the European Commission (EC) and the Netherlands Enterprise Agency (RVO). The authors would like to acknowledge the contributions of A. Ibrahim of the TDT group for is help with the IJTAG implementation.

References

1. S. Davidson, Towards an understanding of no trouble found devices, *IEEE VLSI Test Symp. (VTS)* (Palm Springs, California, USA, 2005), pp. 147–152.
2. J. P. Hofmeister, P. Lall, D. Panchagade, N. N. Roth, T. A. Tracy, J. B. Judkins and K. L. Harris, Ball grid array (BGA) solder joint intermittency detection: SJ BIST, *IEEE Aerospace Conf.* (Big Sky, MT, USA, 2008), pp. 1–11.
3. N. Kranitis, A. Merentitis, N. Laoutaris, G. Theodorou, A. Paschalis, D. Gizopoulos and C. Halatsis, Optimal periodic testing of intermittent faults in embedded pipelined processor applications, *IEEE Design, Automation and Test in Europe (DATE)*, Vol. 1 (Munich, Germany, 2006), pp. 1–6.
4. H. G. Kerkhoff and H. Ebrahimi, Detection of intermittent resistive faults in electronic systems based on the mixed-signal boundary-scan standard, *IEEE Asia Symp. Quality Electronic Design (ASQED)* (Kula Lumpur, Malaysia, 2015), pp. 77–82.
5. H. Ebrahimi, A. Rohani and H. G. Kerkhoff, Detecting intermittent resistive faults in digital CMOS circuits, *IEEE Int. Symp. Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFT)* (Storrs, CT, USA, 2016), pp. 87–90.
6. J. Pan, A control-chart-based method for solder joint crack detection, *J. Microelectron. Electron. Packag.* **11** (2014) 94–103.
7. D. Kwon, M. H. Azarian and M. Pecht, Nondestructive sensing of interconnect failure mechanisms using time-domain reflectometry, *IEEE Sens. J.* **11** (2011) 1236–1241.
8. F. Loete and C. Gilbert, Diagnostic of connector's degradation level by frequency domain reflectometry, *IEEE Holm Conf. Electrical Contacts* (Portland, OR, USA, 2012), pp. 1–4.
9. J. P. Hofmeister, S. Vohnout, C. Mitchell, F. O. Heimes and S. Saha, Halt evaluation of SJ BIST technology for electronic prognostics, *IEEE AUTOTESTCON* (Orlando, FL, USA, 2010), pp. 1–7.

10. B. Steadman, F. Berghout, N. Olsen and B. Sorensen, Intermittent fault detection and isolation system, *IEEE AUTOTESTCON* (Salt Lake City, UT, USA, 2008), pp. 37–40.
11. S. Das, C. Tokunaga, S. Pant, W. Ma, S. Kalaiselvan, K. Lai, D. M. Bull and D. T. Blaauw, Razorii: *In situ* error detection and correction for pvt and ser tolerance, *IEEE J. Solid-State Circuits* **44** (2009) 32–48.
12. L. Lai, V. Chandra, R. C. Aitken and P. Gupta, SlackProbe: A flexible and efficient *in situ* timing slack monitoring methodology, *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* **33** (2014) 1168–1179.
13. M. Sadi, L. Winemberg and M. Tehranipoor, A robust digital sensor IP and sensor insertion flow for *in-situ* path timing slack monitoring in SoCs, *IEEE VLSI Test Symp. (VTS)* (Napa, CA, USA, 2015), pp. 1–6.
14. O. S. Unsal, J. W. Tschanz, K. Bowman, V. De, X. Vera, A. Gonzalez and O. Ergin, Impact of parameter variations on circuits and microarchitecture, *IEEE Micro* **26** (2006) 30–39.
15. H. G. Kerkhoff and H. Ebrahimi, Investigation of intermittent resistive faults in digital CMOS circuits, *World Scientific J. Circuits, Syst. Comput. (JCSC)* **25** (2016) 1640023.
16. W. Shan, L. Shi and J. Yang, *In-situ* timing monitor-based adaptive voltage scaling system for wide-voltage-range applications, *IEEE Access* **5** (2017) 15831–15838.
17. M. Saliva, F. Cacho, V. Huard, X. Federspiel, D. Angot, A. Benhassain, A. Bravaix and L. Anghel, Digital circuits reliability with *in-situ* monitors in 28 nm fully depleted soi, *IEEE Design, Automation and Test in Europe (DATE)* (Grenoble, France, 2015), pp. 441–446.
18. 1687–2014 — IEEE standard for access and control of instrumentation embedded within a semiconductor device (2014).
19. H. Ebrahimi and H. G. Kerkhoff, Intermittent resistance fault detection at board level, *IEEE Int. Symp. Design and Diagnostics of Electronic Circuits & Systems (DDECS)* (Budapest, Hungary, 2018), pp. 135–140.
20. JTAG Live Studio, available: http://www.jtaglive.com/.
21. H. Ebrahimi and H. G. Kerkhoff, Testing for intermittent resistive faults in CMOS integrated systems, *IEEE Euromicro Conf. Digital System Design (DSD)* (Limassol, Cyprus, 2016), pp. 703–707.
22. A. Osborne, *An Introduction to Microcomputers Volume 1: Basic Concepts* (Osborne-McGraw Hill, Berkeley California 1980, USA), ISBN 0-931988-34-9.
23. PI Block Guide V03. 06, Freescale Semiconductor (2003).