Off-line Testing of Delay Faults in NoC Interconnects

Tomas Bengtsson[1], Artur Jutman[2], Shashi Kumar[1], Raimund Ubar[2], Zebo Peng[3]
[1]*Jönköping University, Sweden*
[2]*Tallinn University of Technology, Estonia*
[3]*Linköping University, Sweden*

Abstract

Testing of high density SoCs operating at high clock speeds is an important but difficult problem. Many faults, like delay faults, in such sub-micron chips may only appear when the chip works at normal operating speed. In this paper, we propose a methodology for at-speed testing of delay faults in links connecting two distinct clock domains in a SoC. We give an analytical analysis about the efficiency of this method. We also propose a simple digital hardware structure for the receiver end of the link under test to detect delay faults. It is possible to extend our method to combine it with functional testing of the link and adapt it for on-line testing.

1. Introduction

Many research groups have proposed Network on Chip (NoC) architectures with structured and scalable topologies as an option for implementing multi-core systems on chip (SoC) [2][3]. In order to avoid clock skew and clock distribution problems NoC systems are likely to use a Globally Asynchronous Locally Synchronous (GALS) approach [4]. In a design using a GALS approach, a system is partitioned into multiple domains, each having a different clock.

The job of NoC testing can be divided into three main parts, testing of cores, testing of routers and testing of the interconnecting links between routers. In this paper, focus is on testing of interconnection links (buses) in a NoC

Timing requirements and delay constraints are changing with technology, resulting in the situation where gate delays become less important than transmission wire delays. The delay of a conducting wire may change from an acceptable range to an unacceptable value due to a material defect or variations in the fabrication process. The effect of these factors is enhanced due to shrinking dimensions in sub-micron CMOS technology. The delay could also be affected due to effect of crosstalk from other signal wires. These delay faults are more likely to occur in the high speed links connecting two GALS clock domains. Many of these faults may not be detected if the chip is tested at lower than the operating frequency. At-speed testing of a chip involves both the test generation and the test application challenges. The main concern of the former one is to provide efficient test stimuli of high fault coverage. The main challenge associated with test application is the activation of the worst case behavior of the chip under test and efficient registration of its malfunction. For a NoC test solution to be efficient, both these challenges must be solved by a BIST (Built In Self Test) hardware.

The at-speed test application and measurement scheme for delay faults is rather different in different application areas. Hence, the well-developed theory and methodology of delay testing of synchronous digital logic [11] cannot be applied to testing asynchronous interconnect links.

In this paper we propose a methodology for at-speed testing of delay faults in links connecting two GALS domains on a chip. These two domains could be two switches in a NoC system.

The current lack of dedicated literature on NoC testing [1] can be explained by the infancy of the topic itself as well as by the lack of standardization in NoC architectures. Fortunately, interconnect testing has been an area of extensive research and standardization with respect to PCB (Printed Circuit Board) based systems. Many concepts of PCB interconnect test can be adopted by the NoC paradigm. Since the main concern of current paper is the test response measurement mechanism, we have to look at the response analysis methods in the first place.

The papers [5][6][7] concentrate on testing of crosstalk and noise induced errors in on-chip interconnects. All the papers propose original hardware solutions. A simple design of special crosstalk detection cells is proposed in [5] and [6]. The authors of [5] use a double-sampling principle in BIST components, which is unfortunately overly conservative with the unacceptable false detection level of 40 percent. An alternative design for noise and skew detectors is proposed in [6]. The authors, however, do not provide any data on fault detection levels, which makes their results inconclusive. An expensive hardware solution for accurate delay and crosstalk measurement is proposed in [7]. However, in most cases simple detection and diagnosis of defects (which can be done using much cheaper hardware) is enough.

In [8] another BIST solution with similar complexity is proposed including an aliasing-free response analysis. Another interconnect BIST solution with a similar complexity was proposed in [8]. In [13] this method is used for testing synchronous NoC links.

Note that none of the considered papers except [5] and [10] address the asynchronous NoC interconnects. In [10] we proposed a method for test of asynchronous handshaking links. The method in the current paper is an improved technique similar to that in [10].

2. Preliminaries and Problem Definition

In NoC based architectures, unidirectional physical channels are preferred between two routers or between a core and a router for communication [9]. Most such systems follow the GALS approach. A consequence of this approach is that two routers cannot have a common interpretation of time that is accurate enough to measure delay in the channel between them. The communication channel between two routers will use handshaking protocol.

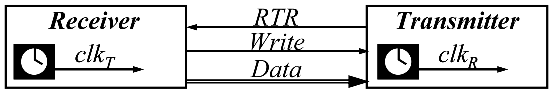

Fig 1. Handshaking protocol

Fig 1 shows data and control connections for a typical handshaking based communication protocol. The handshaking protocol generally has two control signals, *Write* and *Ready-To-Receive* (*RTR*). Along with these there are data lines carrying the information that should be transferred. The receiver asserts RTR=1 when it is ready to receive new data. Signal *RTR* may only rise when *Write* is low and sink when *Write* is high. Signal *Write* may only rise when *RTR* is high and sink when *RTR* is low.

The handshaking protocol together with the clock signals is shown in Fig 2, where signal clk_T is the clock signal at the transmitter and clk_R is the clock signal at the receiver. We assume that a separate clock generator is used within each domain. For efficiency reasons, many transmitter implementations do not give one full clock period gap between asserting the new data and the *Write*.

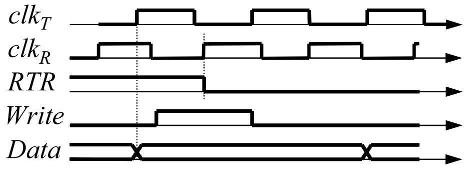

Fig 2. Handshaking in GALS

A delay fault could make a line either slower or faster than allowed. For testing the delay faults in the links, we assume the transmitter and the receiver are fault free. A serious fault situation occurs when one or more of the data lines have larger delay than the signal *Write*. Then there is a risk that a change in the signal *Write* arrives at the receiver before the change in some data line has arrived at the receiver. The effect is that the receiver might read old values of some data bits. Other delay faults are either harmless or degrades the throughput.

In this paper we deal with test for delay faults that might make data erroneous, which is the harder fault to test. We assume that a handshaking protocol as the one described above is used for communication.

3. Proposed Test Method

In this section we first consider one data line and the signal *Write* to illustrate how delay faults can be detected. Fig 2 shows how data transfer works with a handshaking protocol when no delay fault is present. The signal *Write* arrives at the receiver after the arrival of data. The data is then read on the succeeding active clock edge. If there is a delay on data lines the receiver might read old values on these lines. Then erroneous data is read. Let t_l be the time from the data arrives till the signal *Write* arrives at the receiver. Due to delay faults and process variations, this parameter will vary between different devices. If t_l is smaller than zero there is a delay fault that might result in an erroneous data. If t_l is positive, there is no such delay fault affecting the data line under consideration.

Because there is no synchronization between the clocks in the transmitter and receiver, the new data as well as *Write* can arrive in any part of the clock cycle of the receiver. The effect of this is a non-deterministic time gap from the arrival of signal *Write* until the subsequent active clock edge in the receiver. Let T_R denote the clock period time of the receiver. Then the time gap from *Write* arrives till the data is actually read, is in the interval [0, T_R] with uniformly distributed probability. Due to this non-deterministic time gap, the data may be received sometimes correctly and sometimes erroneously when a delay fault exists in the data lines.

The link can be tested off line by sending data which the receiver knows. An erroneous read of the data implies that there is a fault in the link. If the data is read correct it is, however, not possible to say if the delay fault is present or not. A method we can use instead is to read the data on the active clock edge at the receiver just before the arrival of signal *Write* [10]. In such a way the data cannot be read correctly if the delay fault we are looking for is present. If this delay fault is absent data might be read correctly. Fig 3 shows how this works. Somewhere in the shaded area an active edge of the clock of the receiver will be present. At this clock edge we read the data and if this read happens after the data has arrived, it will be read correct. If the delay fault is present, the data arrives after signal *Write* and then correct data will never be read correctly at this clock edge.

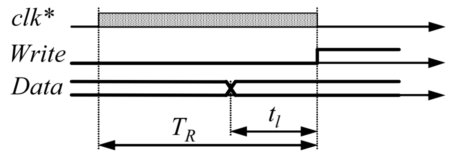

Fig 3. Timing diagram of extra reading

In the proposed method we let the receiver read the data both on its clock edge just before signal *Write* has arrived and on the next clock edge after the arrival of *Write*. The former read is as in the method presented in [10] and the latter is as in the normal operation. We call this procedure an instance of a test. If the data read after arrival of signal *Write* is erroneous or the data read before signal *Write's* arrival is correct, then we are sure that the chip is faulty, or it does not have the delay fault we are looking for respectively. If neither of these cases occur this instance of the test cannot say if this delay fault is present or not.

Let $g(t_l)$ denote the probability that one single test instance can decide (unambiguously) if the fault is present or not, given a t_l. This function is then:

$$g(t_l) = \begin{cases} |t_l|/T_R & |t_l| \le T_R \\ 1 & |t_l| > T_R \end{cases} \tag{1}$$

To test if the fault is present or not, the same instance of the test is repeated until either the chip could be judged or until a predefined number of instances of the test have been made.

4. Hardware Implications of the Method

Fig 4 shows a schematic of the required BIST hardware on the receiver side. We assume, for the sake of simplicity, that testing is initiated by a signal *test_mode* from the transmitter. After receiving this signal Data is continuously sampled at every active edge of clock. The two latest samples are stored. After sampling a new data after the arrival of *Write,* the contents of these registers are compared with the expected data. The comparison results are used to decide whether to abort testing or to try another instance of the test or to declare the tested fault as absent.

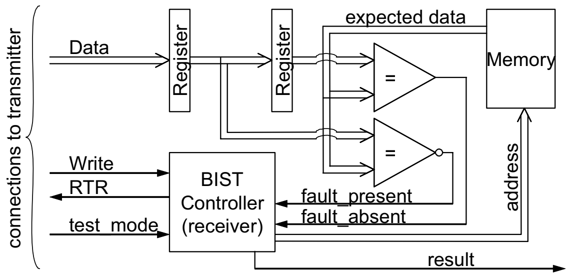

Fig 4. Receiver

It should be noted that all these BIST hardware resources may not be required in actual practice. Already available hardware resources, like registers and memory buffer, in the switch can be shared for BIST purposes. Also, the same BIST hardware can be used for different links connecting switches in a time multiplexed manner. The BIST hardware at the transmitter is simpler than at the receiver.

5. Analysis and Results

Variable t_l will depend on process variations and other manufacturing effects. This means that t_l is a stochastic variable. We assume that t_l has a normal distribution with an expected value μ and a standard deviation σ. The nominate value on t_l is then μ. Given μ and the probability that the delay fault we are looking for is present, σ can be determined. Let $f(t)$ denote the density function of t_l.

Let p denote the probability that one instance of the test can detect if the delay fault is present or not. But p itself is a stochastic variable, because it depends on the outcome of t_l. Let $r(x)$ denote the density function of p.

Let a and b denote two real numbers such that $a < b$. The following relation exists between $r(x)$, $g(t_l)$ and $f(t)$:

$$\int_a^b r(x)dx = \int_{a<g(t)<b} f(t)dt \qquad \forall(a,b)\ ,a<b \tag{2}$$

It is possible to show that to conclusively prove that a device is correct or faulty, the number required test will be infinite on average. The consequence of this is that we need to terminate our test after a maximum number of tries (say l) with a possibility of non-conclusive result. Given a percentage of non-conclusive results that we can accept, the value of l can be computed. Given l, let the stochastic variable p_l be the number of instances of a test actually performed during a test. Let $q_l(n)$ denote the density function for p_l. Let $h_l(p)$ be the expected value of p_l given l and p.

$$h_l(p) = \sum_{i=-\infty}^{+\infty} i \cdot q_l(i) = \left(\sum_{i=1}^{l-1} i \cdot (1-p)^{i-1} p \right) + l \cdot (1-p)^{l-1} \quad (3)$$

Let n_b denote the expected value of $h_l(p)$. This is the expected number of instances of a test that will be performed for test of a delay fault.

$$n_b = \int_{-\infty}^{+\infty} r(x) \cdot h_l(x) dx \quad (4)$$

To demonstrate the efficiency of the proposed method we have computed n_b for three different values of each of the following two parameters: (i) ratios between μ (the average value of t_l) and T_R and (ii) probability of the fault in tested link. The value l, is chosen as $l = p_f$ / 10. Table 1 shows the average number of tests required n_b. Figures in brackets in this table shows the number of tests needed if method in [10] is used.

Table 1. Average no of tests required

	Fault probability P_f		
μ/T_R	*0.1*	*0.01*	*0.001*
1.0	*2.6 (6.3)*	*1.8 (2.8)*	*1.4 (1.7)*
0.5	*4.8 (13)*	*3.3 (5.2)*	*2.6 (3.3)*
0.1	*24 (62)*	*16 (27)*	*13 (15)*

Number of test instances required is larger for smaller nominate values of t_l and for chips with higher probability of delay faults in links. The reason in both cases is that there is a larger probability mass for values on t_l closer to zero. This makes the probability, that a test instance can judge if a fault is present or not, smaller.

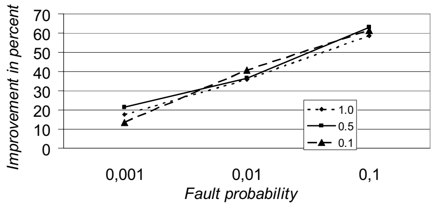

Fig 5. Comparison with previous method [10]

In [10] we presented a method in which the data was read only once using a clock edge just before signal *Write* arrives. This method is inefficient if the probability for fault is high. Fig 5 shows efficiency improvements in percentage using the method presented here over the method in [10]. There are three curves representing three different values of t_l/T_R. As expected the improvement is larger when the probability for fault is large.

6. Conclusions

We have proposed a simple methodology for off-line testing of delay faults in links connecting two clock domains in SoCs built using GALS principles. We have shown that our method is efficient and on average require a small number of test trials to label a chip good or faulty. Our method has one limitation that in the worst case, a finite number of test trials may not be able to label a chip correct or faulty. In such a case, we discard the unlabelled chips. The probability of this situation can be made arbitrarily small by increasing the number of test trials. In other words, our method offers a trade-off between test time and number of good chips which are discarded. An advantage with our test methodology is that it requires only pure digital test hardware. It should be quite easy to combine our method of delay fault testing with existing methods of functional testing of interconnection links [8]. The methodology can also be easily extended for on-line testing.

References

[1] A. Jantsch and H. Tenhunen (Editors), "Networks on Chip", *Kluwer Academic* Publishers, 2003.

[2] L. Benini, G. De Micheli "Networks on Chips: A New SoC Paradigm", *IEEE Comp*. Vol.35 no.1, Jan. 2002

[3] S. Kumar et. al., "A Network on Chip Architecture and Design Methodology", *IEEE Comp. Society Annual Symp. on VLSI,* Pittsburgh 2002, pp. 117-124.

[4] A. Hemani et. al., "Lowering power consumption in clock by using Globally Asynchronous Locally Synchronous Design Style", *Proceedings of DAC*, 1999, pp 873-878.

[5] Y.Zhao et al. "Double sampling data checking technique: an online testing solution for multisource noise-induced errors on on-chip interconnects and buses", in *IEEE Trans on VLSI,* vol. 12, no. 7, July 2004, p.746-755.

[6] A. Attarha, M. Nourani, "Testing Interconnects for Noise and Skew in Gigahertz SoC", *in Proc. of ITC*., 2001, pp. 305-314.

[7] C.Su, et al, "All Digital Built-in Delay and Crosstalk Measurement for On-Chip Buses", *in Proc. of DATE Conf*., March 27-30, 2000, pp.527 - 531.

[8] A. Jutman, "At-Speed On-Chip Diagnosis of Board-Level Interconnect Faults", in *Formal Proc. of 9th Euro¬pean Test Symposium*, France, May 2004, pp. 2-7.

[9] D. Pamunuwa, "Modelling and Analysis of Interconnects for Deep Submicron SoC", *PhD Thesis, Royal Institute of Technology*, Stockholm 2003.

[10] T. Bengtsson, A. Jutman, S. Kumar and R. Ubar, "Delay testing of asynchronous NoC interconnects", in *Proceedings of 12th MixDes Conference*, June 2005

[11] Angela Krstic, Kwang-Ting (Tim) Cheng, "Delay Fault Testing for VLSI Circuits", 1998, 208 p.

[12] W. Stallings, Data and Computer "Communications, 7th ed", *Prentice Hall*, 2004, 864 p.

[13] A.Jutman, R.Ubar, J.Raik, "New Built-In Self-Test Scheme for SoC Interconnect", in *Proc of 9th WMSCI*, Orlando, Florida, USA, July 10-13, 2005, vol.4, pp.19-24.