Fault Localization in NoCs by Timed Heartbeats

Bernhard Fechner, Arne Garbade, Sebastian Weis, Theo Ungerer
Department of Computer Science
University of Augsburg
Universitaetsstr. 6a
D-86159 Augsburg
{fechner, garbade, weis, ungerer}@informatik.uni-augsburg.de

Abstract: Future computing systems will contain more and more cores on a single die. Permanent faults occur not only during manufacturing but may also arise at runtime. To detect these faults, a group of cores is monitored by a single unit, receiving heartbeats from all cores. In this paper, we present a simple method to localize permanent faults in a 2D mesh-based NoC by using heartbeats and by measuring the time from source (core) to destination (monitoring unit). We introduce a heartbeat network along with the normal application message network to guarantee a deterministic heartbeat timing and no interferences with application messages. If the time for a heartbeat exceeds a given interval, it can be concluded that the heartbeat is missing or delayed, e.g. because of a faulty core, link or router. As this is not sufficient to localize a fault, we introduce the concept of Timed Heartbeats, which uses different routing directions in contrary to the intended routing to introduce a fixed, additional delay for rerouted heartbeats. The delay helps to localize the fault without any additional bandwidth consumption.

1 Introduction

Advances in VLSI technology enable to integrate a high number of cores on a single die, communicating over an interconnection network. Examples are the 64-core TILE64 from Tilera [BGH86], the experimental Intel Polaris 80-core [Dii09], or the recently presented 1024-core accelerator Rigel [JJK+11]. All of these many-core chips include packet-based Network-on-Chips (NoC) in order to scale with the number of cores and reduce the hardware complexity of the processor. In particular, 2D mesh topologies are currently popular because they combine scalability with low design complexity and support simple dimensional ordered routing algorithms. Therefore, we take the 2D mesh as our baseline topology.

Due to shrinking feature sizes, rising chip temperatures etc. more and more permanent, intermittent, and transient faults in cores, routers, and links can occur. These can have many manifestations—not only at manufacturing time but mainly during operation. We assume permanent link and router faults and that the end-to-end communication within the application message network (AMN) is secured against transient faults, i.e. safeguarded with error detecting/correcting codes (EDC/ECC).

To check whether a component (core or router) of a many-core system is alive, we assume a monitoring architecture similar to [WGSU11], where periodic heartbeats are sent from each component to a monitoring unit, called Fault Detection Unit (FDU). The FDU is the abstraction of a monitoring unit. It can be implemented either as a complex circuit with the ability to dynamically analyze the information gathered from the monitored components (running sophisticated algorithms) and starting actions to maintain the functionality of the chip or (as in this work) as simple unit collecting information about faults.

The proposed Timed Heartbeat technique uses the arrival times of periodically sent heartbeats to localize link and router faults. By comparing the expected (deterministic) arrival time with the actual arrival time, it can be concluded if a heartbeat was rerouted due to a fault or not. A simple switching of the routing algorithms allows to localize the fault. Please note, that this technique can be applied to any topology as long as arrival times of heartbeats are deterministic.

We introduce a heartbeat network (HN) along with the normal application message network (AMN) to guarantee that the heartbeat timing is deterministic and heartbeats do not interfere with the much larger application messages. In this work, we mean the HN if not otherwise stated. In comparison with a standard packet-based communication, we do not have any additional traffic overhead (packet header, fixed packet length, EDC/ECC) and therefore can save bandwidth $((N-1)*(length\_of\_packet-1))$ and power each network cycle.

This paper is structured as follows: Section 2 discusses related work and Section 3 provides a description of the basic network architecture. Section 4 presents the localization of faults with Timed Heartbeats. Section 5 concludes the paper. For the convenience of the reader, we provide a list of abbreviations in Table 1.

Abbreviation	**Meaning**
AMN	Application Message Network
(B)MQ	(Bandwidth) Multi-Quadrant
(B)SQ	(Bandwidth) Single-Quadrant
EAT	Expected Arrival Time
FDU	Fault-Detection Unit
HN	Heartbeat Network
MAT	Measured Arrival Time
MD, d$_m$	Manhattan Distance
NoC	Network on Chip
QHC	Quadrant Heartbeat Cycle

Table 1: List of abbreviations

2 Related work

Heartbeats are a well-known timing-based mechanism to detect faults in distributed systems [CT96, BMS02, SPTU07, SND11]. Examples can be found in many early commercial fault-tolerant systems such as Tandem [BGH86] and later in the Globus Heartbeat Monitor [SFK+98]. Details on routing problems and algorithms can be found in [GHKS98]. More details on heartbeats and timing intervals in multicomputers can be found in [HS98].

Steinert and Gillblad [SG10] recently proposed to use collaborating nodes to locate a fault in a distributed system. They applied statistical methods to compute the expected timing from measured network delays. However, this requires the costly evaluation of a probability density function. In contrast to fault localization in distributed systems, our work exploits the determinism of on-chip interconnects for a much simpler localization of link and router faults.

3 Basic network architecture

As topology we assume a (n, m) 2D-mesh, where $N = n * m$ is the number of cores. For simplicity, we assume the mesh as quadratic $(m = n)$. Let $G = (V, E)$ be a graph G with a set of vertices V and edges E. The number of cores N is given by $N = |V|$. Since one router is directly associated to a core, the number of routers is $R = N$. The number of edges is $|E| = 2(N - \sqrt{N})$. The mesh has a diameter of $2\sqrt{N} - 1$ and a maximal degree of 4. The mesh (s. Figure 1 and 2) is divided in four quadrants (North-West [NW], North-East [NE], South-West [SW], South-East [SE]), called *multi-quadrant* (MQ) and four quadrants North [NO], South [SO], East [EA], West [WE], called *single-quadrant* (SQ). We regard one MQ (NW) and two SQs (NO, WE) since all other quadrants can be handled analogously. The communication of the core to the router and vice-versa is accomplished via a local link. The router has four additional ports (North [NO], South [SO], East [EA], West [WE]) for the communication to its neighbors. Links are assumed as bidirectional. Several possibilities concerning the communication times can be made, since there are four neighboring nodes and two wire delays (Δx, Δy), introducing different skews between nodes. We assume the simplest case where all wire delays are (for every direction) equal to one (Δx=Δy=1). Since the FDU resides in the center, the transmission delays are equal to the Manhattan distance $d_M(x, y) = |x| + |y|$.

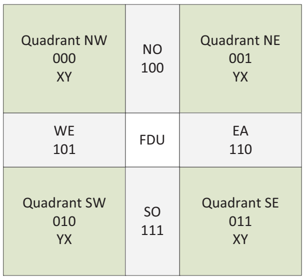

Figure 1: Coding of Quadrants

3.1 Sending of Heartbeats

All cores send heartbeats to the FDU residing in the center of all quadrants (s. Figure 2). The localization of a faulty link or router is simple, if we allow to send the ID of a faulty link or router to the FDU, complicating the router design. We assume that the FDU (router) is able to handle one heartbeat at a time. It is possible to handle more heartbeats at a time, since the FDU can be implemented effectively by using a single decrementer and a table holding all excepted/measured arrival times.

The heartbeats propagate each network clock cycle from router to router. A heartbeat transmission is always initiated from a core via the local router with the FDU as destination. It can only be sent if all local links (one for the heartbeat and the ones for the AMN) are fully functional. The links are considered as functional, iff no fault occurred before. The router therefore has a counter for each link, holding the number of detected faults in the AMN. Additionally, we can measure the time from one fault to another, e.g. to relate the number of faults in time to be able to detect permanent faults.

3.2 Bandwidth composition, pure XY routing

Regarding XY-routing for heartbeat transmission, the heartbeat is first routed in X direction (dimension 0) until it reaches the destination column, then in Y direction (dimension 1). Each router has a field, which determines the quadrant relative to the FDU. The location field is used for routing decisions. Figure 1 shows the coding scheme for each quadrant of the network. The first bit determines if the quadrant is a MQ (set to 0) or SQ (set to 1). Figure 2 shows details about fault location, mesh assembly, distances and quadrants.

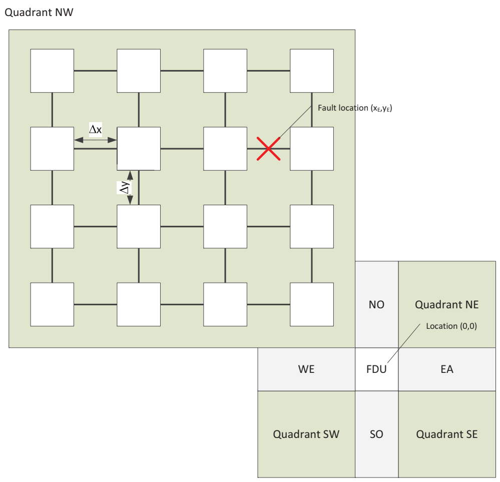

Figure 2: Quadrants, mesh assembly and distances

Routers located in the SQs NO and SO are the only ones routing in Y direction. All others route in X direction. Since we have one FDU and a quadratic mesh, the maximum number of cores sending to the FDU concerning each MQ/SQ can be easily computed. For SQs WE, EA, NO, SO: $BSQ = \frac{\sqrt{N}-1}{2}$. For MQs NE, NW, SE, SW: $BMQ = BSQ * BSQ = \frac{1}{4}(\sqrt{N}-1)^2$. The total bandwidth of heartbeats in relation to each router input (R_{NO}, R_{EA}, R_{SO}, R_{WE}) results to $R_{SO} = R_{NO} = 2BMQ + BSQ$, $R_{EA} = R_{WE} = BSQ$. We check the total number of cores: $4BMQ + 4BSQ = (\sqrt{N}-1)^2 + 2\sqrt{N} - 2 = N - 1$. Obviously the number of heartbeats is unfairly distributed over the SQs (NO, SO). The solution is to alternate between XY and YX routing (s. Figure 1) for different quadrants to distribute the number of heartbeats over all SQs.

4 Fault localization in a NoC with Timed Routing

4.1 Receiving timing information

A fault can either occur at a router or link. We assume that a permanent (link or router) fault can be detected by the router or a neighboring router due to e.g. high impedance of the link or the methods briefly sketched in Section 3. If a link fault has been detected, the appropriate router will reconfigure itself to route around the fault. In subsection 4.4 we describe, how this can be accomplished. The FDU receives the heartbeats and compares the arrival times with the stored timing information, which is equal to the Manhattan distance (MD). The heartbeat send pattern ensures that we have no conflicts, i.e. not more than a single heartbeat arrives at a router. The heartbeat pattern is like a rope of pearls arriving at the FDU, first heartbeats of SQs then heartbeats of MQs starting from the bottommost (MQ NW) to the topmost row. To ensure that we have no conflicts, a delay must be introduced from row to row. The tables $EAT_{\{NW,NO,NE,EA,SE,SO,SW,WE\}}$ in the FDU hold the expected arrival times for heartbeats for each quadrant. For example, the table for the quadrant EAT_{NW} is $\begin{pmatrix} n^2 & \dots & n \\ \vdots & \ddots & \vdots \\ n & \dots & 1 \end{pmatrix}$, whereas the following matrix holds the times when heartbeats are sent: $\begin{pmatrix} n^2 & \dots & n^2-n+1 \\ \vdots & \ddots & \vdots \\ n & \dots & 1 \end{pmatrix}$. The FDU associates the heartbeat to a table entry of the measured arrival times (MAT) according to its specific arrival time. The value in the MAT is then decremented. We must therefore conduct four subtractions in parallel in each network clock cycle. Since the results with equal MD are the same, we need to compute it once per quadrant table for a single MD and distribute it to the other tables, if all heartbeats arrived. If a heartbeat arrives too late, the according value will be less than zero. Alternatively, we can wait until all heartbeats arrived, then XOR the MAT with the EAT, $F = EAT \oplus MAT$. The fault matrix F signals any differences between the expected and measured arrival times, iff $F \neq 0$. In the following, a *deviation* means a deviation in the timing of a single heartbeat. To generate the heartbeat at a specific time, only a single decrementer is needed. If the decrementer triggering the heartbeat is defect (permanent or transient), this will lead to a deviation. Therefore, the decrementer must not be protected against faults.

4.2 Localization of faulty links

On link-level, a fault is assumed to occur permanently at position (x_L, y_L). Note, that we still regard the MQ NW.

Horizontal link fault (XY-routing): If a link fault occurs at position (x_L, y_L), obviously no router above $> y_L$ or below $< y_L$ the fault position will have to reroute, not leading to any deviation. Therefore the row in which the fault occurred can be localized perfectly. All (fault-free) links with position $< x_L$ will not lead to a deviation. All heartbeats initiated leftwards $\geq x_L$ will arrive later. The fault matrix

$$\begin{pmatrix} 0 & \dots & 0 \\ \vdots & \ddots & \vdots \\ 1 & \dots 1 & 0 \end{pmatrix}$$

illustrates this. The most rightwards "1" is the router with the faulty link, since it signals the first occurrence of the fault and thus the first deviation. Naturally, the fault can be perfectly localized in this case, but in the worst case, we must wait BSQ network cycles. XY-routing does not provide information about faulty links in Y direction in the MQ NW (other quadrants analogously). Therefore, the routing algorithm is switched from XY to YX routing after all heartbeat messages of a quadrant arrived (a quadrant heartbeat cycle, QHC).

Vertical link fault (YX-routing): Obviously no router with position left ($> x_L$) or right ($< x_L$) of the fault will have to reroute, not leading to any deviation. Therefore the column in which the fault occurred can be localized. All links below or above the fault ($< y_L$, $\geq y_L$) will lead to an increased timing. The fault matrix

$$\begin{pmatrix} 0\dots & 1 & 0 \\ \vdots & \ddots & \vdots \\ 0\dots & 1 & 0 \end{pmatrix}$$

depicts this.

4.3 Localization of faulty routers

A router fault is modeled by a maximum of five concurrently occurring link faults (note, that core and router faults are not distinguishable and we therefore do not regard faulty local links). The router is assumed to exhibit a fail-stop behavior. Thus, it is not able to send or receive messages any more after a fault. Concerning XY-routing two link faults have to be regarded (WE and EA, YX analogously). A single router failed, if the FDU receives only $N-2$ heartbeats in a given time interval. If there is a faulty router at position (x_R, y_R), no router above $> y_R$ or below $< y_R$ the fault position will have to reroute (due to the assumed XY-routing). As for link faults, the row in which the fault occurred can be localized. All cores located right the fault $< x_R$ are able to produce heartbeats. All heartbeats from cores leftwards $\geq x_R$ are rerouted and introduce an additional delay. Since the routing algorithm is switched from XY to YX routing, we detect two faulty links in two successive QHCs and missed one heartbeat in each QHC. The following fault matrix illustrates the situation of a router fault in the bottommost row after two QHCs:

$$F_{bottommost} = \begin{pmatrix} 0\ldots & 1 & 0 \\ \vdots & \ddots & \vdots \\ 1\ldots & 1 & 0 \end{pmatrix}.$$

We can also distinguish router or link faults in the topmost row of the mesh. Since the routing algorithm is switched from XY to YX routing, we detect faulty horizontal links in one QHC and missed one heartbeat in each QHC. F depicts the situation after two QHCs:

$$F_{topmost} = \begin{pmatrix} 1\ldots & 1 & 0 \\ \vdots & \ddots & \vdots \\ 0\ldots & 0 & 0 \end{pmatrix}.$$

4.4 Rerouting around a faulty link

Until now, we did not specify, how to reroute around a faulty link. There can be several routing possibilities. Figure 3 shows the possibilities for a link fault rightwards core S for a heartbeat propagating to core T. We distinguish two main cases:

1. it is allowed to route back in the direction the heartbeat came from: (s. Figure 3) if we allow to reroute back (route 3), in the MQ NW (XY routing QHC), we have two possibilities, route westwards (WE, route 3), then to the north (NO, route 4, delay +4) or south (SO, route 3, delay +2). Since the router receives a heartbeat from an unexpected direction (EA instead WE), it knows that a fault occurred and that it has to reroute to a different location (not in the direction of the fault, EA).
2. if this is disallowed: the heartbeat can be rerouted in such a way that no delay occurs (route 1, SO) or with route 2, delay +2, NO.

Port uesd as input	i
Port used as output	o
Port has permanent fault	x
Delay introduced by rerouting	(+j)

Table 2: Notation used in Table 3

Table 3 lists the routing combinations for all quadrants, whereas we use the notation introduced in Table 2.

Type	**Quadrant**	**From**	**To**	**Fault**	**Reroute**	**Reroute back**
MQ	NW	iWE	oEA	xNO	–	–
		iWE	oEA	xSO	–	–
		iWE	oEA	xEA	NO(+2),SO(+0)	WE,NO(+4); WE,SO(+2)
SQ	NO	iNO	oSO	xWE	–	–
		iNO	oSO	xEA	–	–
		iNO	oSO	xSO	WE(+2),EA(+2)	NO,(WE,EA)(+4)
MQ	NE	iEA	oSO	xNO	–	–
		iEA	oSO	xSO	–	–
		iEA	oSO	xWE	NO(+2),SO(+0)	EA,NO(+4); EA,SO(+2)
SQ	EA	iEA	oWE	xNO	–	–
		iEA	oWE	xSO	–	–
		iEA	oWE	xWE	NO(+2),SO(+2)	EA,(NO,SO)(+4)
MQ	SE	iEA	oWE	xNO	–	–
		iEA	oWE	xSO	–	–
		iEA	oWE	xWE	NO(+0),SO(+2)	EA,NO(+4); EA,SO(+2)
SQ	SO	iSO	oNO	xWE	–	–
		iSO	oNO	xEA	–	–
		iSO	oNO	xNO	WE(+2),EA(+2)	SO(WE,EA)(+4)
MQ	SW	iWE	oEA	xNO	–	–
		iWE	oEA	xSO	–	–
		iWE	oEA	xEA	NO(+2), SO(+2)	WE,NO(+4); WE,SO(+2)
SQ	WE	iWE	oEA	xNO	–	–
		iWE	oEA	xSO	–	–
		iWE	oEA	xEA	NO(+2),SO(+2)	WE,(NO,SO)(+4)

Table 3: Routing combinations for all quadrants

4.5 Congestions and optimality

The FDU receives heartbeats from $N-1$ cores. Since the FDU router was assumed to accept one heartbeats at a time, optimally in time $N-1$ all heartbeats arrived. Note, that the faster the heartbeats arrive, the faster we are able to locate and handle a fault. Two cases must be distinguished to avoid congestions (assuming that different quadrants are not allowed to simultaneously issue heartbeats):

1. No fault occurred: If we allow to send heartbeats from the bottommost row first, we have to wait row-1 times for each MQ (SQ first).
2. A heartbeat must be rerouted due to a fault: We consider the MQ NW and XY routing. Obviously, no congestion occurs, if we reroute in a direction causing no deviation (s. Figure 3, route 1). This brings no advantage, since we want to locate faults with the introduced delays. Let d_0 be the introduced delay due to a reroute. Then all nodes sending heartbeats with $MD \geq d_0$ will cause a conflict. The solution is simple: We introduce a delay from row to row which is exactly the number of elements in a row.

5 Summary, conclusions and future work

In this paper, we presented a mechanism to localize and distinguish faults on router and link level in NoCs with timing as sole information. Cores are sending heartbeats to the FDU. We extended the existing XY-routing to receive different timings for rerouted messages. We concluded and showed that the position of a fault can be determined. The simple heartbeat network saves bandwidth and energy and enables a faster detection since the heartbeat network can have higher clock rates in comparison with the AMN due to the simple assembly of heartbeats. The method is able to localize four concurrent link and router faults in different MQs perfectly. Furthermore, our technique can be applied to any network topology as long as arrival times of heartbeats are deterministic. Many interesting research opportunities result: what is the relation between different routing algorithms, the localization accuracy and the mean time to detect a fault? How precisely

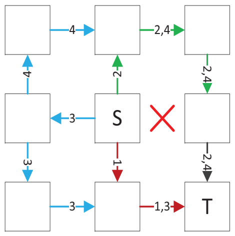

Figure 3: Rerouting around a faulty link

can this combination locate a fault under the influence of multiple faults in a single MQ or SQ? Is there a lower bound concerning the timing for all rerouted messages? Our future work will consider these open questions and also multiple link and router faults.

Acknowledgements. This work was partly funded by the European FP7 project TERAFLUX, ID 249013, http://www.teraflux.eu.

References

[BGH86] J. Bartlett, J. Gray, and B. Horst. Fault Tolerance in Tandem Computer Systems. Technical report, Tandem Technical report 86.2, 1986.

[BMS02] Marin Bertier, Olivier Marin, and Pierre Sens. Implementation and Performance Evaluation of an Adaptable Failure Detector. In *Proceedings of the 2002 International Conference on Dependable Systems and Networks*, DSN '02, pages 354–363, Washington, DC, USA, 2002. IEEE Computer Society.

[CT96] Tushar Deepak Chandra and Sam Toueg. Unreliable failure detectors for reliable distributed systems. *J. ACM*, 43:225–267, March 1996.

[Dii09] S. Diighe. Lessons Learned From The 80-Core Tera-Scale Research Processor. In *Intel Technology Journal*, volume 13, 2009.

[GHKS98] Miltos D. Grammatikakis, D. Frank Hsu, Miro Kraetzl, and Jop F. Sibeyn. Packet Routing In Fixed-Connection Networks: A Survey, 1998.

[HS98] Seungjae Han and Kang G. Shin. Experimental Evaluation of Failure-Detection Schemes in Real-time Communication Networks. In *in Proc. IEEE FTCS*, pages 122–131, 1998.

[JJK$^+$11] D.R. Johnson, M.R. Johnson, J.H. Kelm, W. Tuohy, S.S. Lumetta, and S.J. Patel. Rigel: A 1,024-Core Single-Chip Accelerator Architecture. *Micro, IEEE*, 31(4):30 –41, july-aug. 2011.

[SFK$^+$98] P. Stelling, I. Foster, C. Kesselman, C. Lee, and G. Von Laszewski. A fault detection service for wide area distributed computations. In *High Performance Distributed Computing, 1998. Proceedings. The Seventh International Symposium on*, pages 268 –278, jul 1998.

[SG10] R. Steinert and D. Gillblad. Towards Distributed and Adaptive Detection and Localisation of Network Faults. In *Telecommunications (AICT), 2010 Sixth Advanced International Conference on*, pages 384 –389, may 2010.

[SND11] A. Shikri, M. Noor, and M. Deris. Dynamic-Hybrid Fault Detection Methodology. In *Journal of Computing*, volume 3, pages 58–64, 2011.

[SPTU07] Benjamin Satzger, Andreas Pietzowski, Wolfgang Trumler, and Theo Ungerer. A new Adaptive Accrual Failure Detector for Dependable Distributed Systems. In *ACM Symposium on Applied Computing (SAC 2007)*, pages 551–555, 2007.

[WGSU11] Sebastian Weis, Arne Garbade, Sebastian Schlingmann, and Theo Ungerer. Towards Fault Detection Units as an Autonomous Fault Detection Approach for Future Many-Cores. In *ARCS 2011 Workshop Proceedings (SCAFT Workshop)*, pages 20–23. VDE Verlag, February 2011.