Hindawi Publishing Corporation
VLSI Design
Volume 2007, Article ID 94676, 13 pages
doi:10.1155/2007/94676

Research Article

Online Reconfigurable Self-Timed Links for Fault Tolerant NoC

Teijo Lehtonen,[1,2] Pasi Liljeberg,[2] and Juha Plosila[2,3]

[1] *Turku Centre for Computer Science (TUCS), Joukahaisenkatu 3–5 B, 20520 Turku, Finland*
[2] *Department of Information Technology, University of Turku, 20014 Turku, Finland*
[3] *Research Council for Natural Sciences and Engineering, Academy of Finland, 00501 Helsinki, Finland*

Received 15 October 2006; Accepted 4 March 2007

Recommended by Davide Bertozzi

We propose link structures for NoC that have properties for tolerating efficiently transient, intermittent, and permanent errors. This is a necessary step to be taken in order to implement reliable systems in future nanoscale technologies. The protection against transient errors is realized using Hamming coding and interleaving for error detection and retransmission as the recovery method. We introduce two approaches for tackling the intermittent and permanent errors. In the first approach, spare wires are introduced together with reconfiguration circuitry. The other approach uses time redundancy, the transmission is split into two parts, where the data is doubled. In both structures the presence of permanent or intermittent errors is monitored by analyzing previous error syndromes. The links are based on self-timed signaling in which the handshake signals are protected using triple modular redundancy. We present the structures, operation, and designs for the different components of the links. The fault tolerance properties are analyzed using a fault model containing temporary, intermittent, and permanent faults that occur both as bursts and as single faults. The results show a considerable enhancement in the fault tolerance at the cost of performance and area, and with only a slight increase in power consumption.

1. INTRODUCTION

The move towards nanoscale circuits increases performance and capacities of ICs, but poses new challenges to circuit design. As the dimensions shrink dramatically, it is becoming increasingly difficult to control the variance of physical parameters in the manufacturing process. This results in faults and decreased yield which increases the costs per functioning chip [1, 2]. The yield can be maintained at an acceptable level by admitting some amount of faults in a chip, which then have to be tolerated with dedicated circuit structures. Different types of errors can be tackled using a variety of fault tolerance methods. No single method is sufficient for all types of errors and therefore a sophisticated combination of them is needed [3].

Network-on-chip (NoC) is a structure believed to be the basic platform of future designs [4]. In such a system the communication links between system modules are crucial for the correct operation of system. In this paper, we focus on fault tolerance design of the communication links in NoC architectures. We take into consideration both the permanent and intermittent errors that are commonly originated from the manufacture process or produced by electromigration as well as transient errors caused, for example, by different noise sources and radiation.

The paper is organized as follows. In Section 2 we give background for the presented approach and discuss some related works. The idea is presented in Section 3 followed by the description of the created realizations in Section 4. The results are presented in Section 5 and in Section 6 the fault tolerance properties are analyzed. The power analysis is presented in Section 7, and Section 8 contains discussion about the results and possible enhancements to the design. Finally the conclusions are presented in Section 9.

2. BACKGROUND

An NoC system consists of many processing blocks which have different timing requirements and can operate at different clock frequencies. Communication between these blocks needs synchronization which is error-prone. Also the clock distribution over a wide chip with low skew and jitter is problematic. A viable solution for this is the use of the

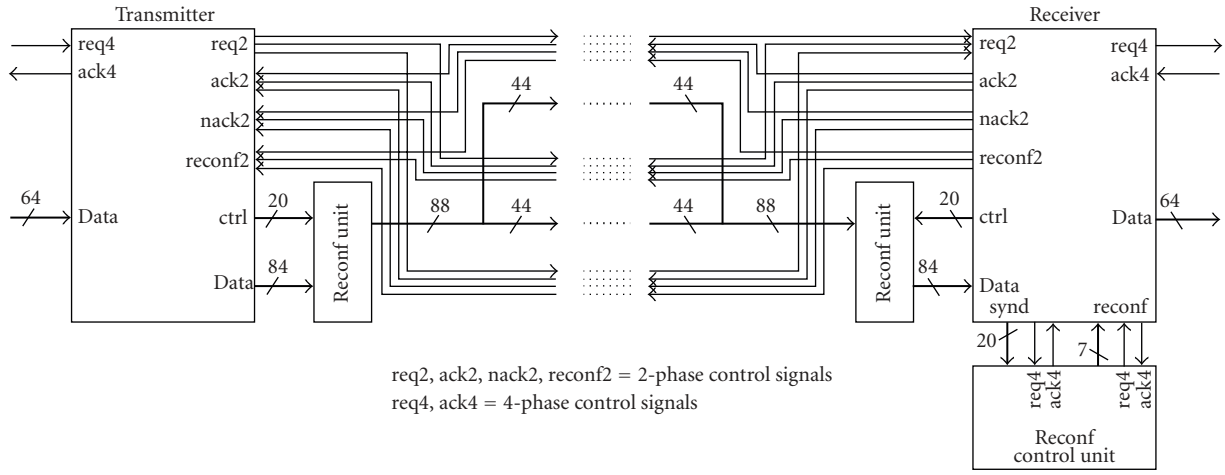


Figure 1: Simplified structure of the proposed link with spare wires and reconfiguration circuits.

globally-asynchronous locally-synchronous (GALS) design approach, where communication between processing blocks is done asynchronously [5–7]. Therefore, we base our link on self-timed design principles [8–10].

When designing a fault tolerant on-chip link for a deep submicron chip, one should first consider the possible fault scenarios. The approach must be capable of tolerating multiple bidirectional errors as well as burst errors [2, 11]. For detecting multiple bidirectional errors the Hamming code is widely used [11–13]. The standard version of it can detect two single errors and with one additional check bit the error detection ability can be extended to three. Cyclic codes can be used to detect burst errors [11]. Another approach for handling burst errors is interleaving [14].

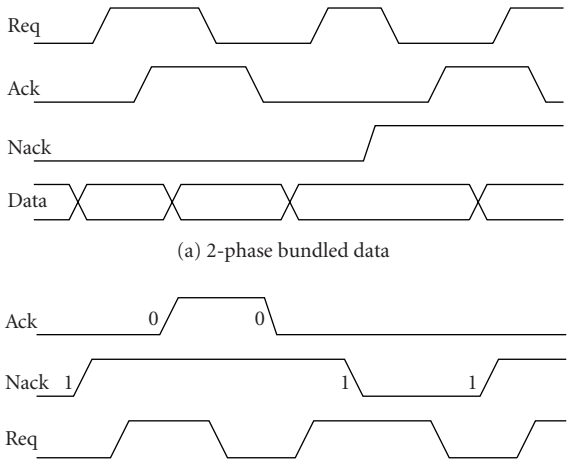
The faults can be categorized into three classes: permanent, intermittent, and temporary or transient faults [1]. Most of the failures (80%) are caused by transient faults [4]. The other way around, up to one fifth of all the failures are originating from permanent or intermittent faults. Thus, the fault tolerance approach must contain elements to not only tolerate the temporary errors but also the ones of more permanent nature. Most of the research has been concentrating on tolerating transient errors. The two methods for this purpose are forward error control (FEC) and automatic repeat query (ARQ) [15], the latter of which is found to be more energy efficient [11]. In FEC the errors are corrected at the receiver based on the information of the check bits while in ARQ the check bits are used to detect errors and a retransmission is requested when necessary. The ARQ will not work in the presence of permanent errors and FEC with commonly used Hamming coding loses its effectiveness already in the presence of a single permanent fault.

Fault tolerance methods besides coding include for instance triple modular redundancy (TMR) and the use of spare components together with error detection [16]. The use of spare wires for NoC interconnects has been presented in [17], where the focus, however, has been on improving yield and no method for fault detection nor reconfiguration protocol is presented.

3. LINK STRUCTURE

The idea in this work is to combine two different fault tolerance methods to achieve a system that is efficient in tolerating transient, intermittent and permanent errors. For transient faults we use Hamming coding and interleaving to detect faults and ARQ as the recovery method motivated by its energy efficiency as reported in [11]. For tolerating permanent and intermittent errors we introduce two methods, one that uses hardware redundancy and the other based on time redundancy. In the first approach spare wires are introduced together with reconfiguration circuits. In the second approach, the data is split into two transmissions and in both of them the transmitted data is doubled. In the receiver the fault-free copy is chosen and the data of the two transmissions are again combined into a whole word. In both structures, the presence of a permanent fault is detected using the same Hamming code as for transient faults. A number of previous error syndromes are stored and if they equal, it indicates a permanent error. The exact error location can be determined by decoding the syndrome. A similar method is used to switch back to the normal operation mode in the split transmission approach. If all the syndromes are zero for the past transmissions, the error has probably been intermittent and change back to the normal mode can be carried out.

The spare wire approach aims at providing unchanged performance in the presence of errors. However, the reconfiguration procedure is allowed to take some time since it is a rare occasion. The split transmission approach on the other hand has a significant impact on the performance and its use is motivated by considering a typical NoC system. An erroneous link could be bypassed through neighboring routers, but this would increase traffic in other links and may result in congestion. Nevertheless, the total latency of links



(a) 2-phase bundled data

(b) 2-phase dual rail (transmitting data "10011")

Figure 2: Self-timed signaling.

and routers on a bypassing route set an objective to the split transmission link design.

The width of the target link is set to 64 bits, which is split into four identical parts. Every part is encoded with Hamming (21, 16) code and the parts are interleaved. In other words, if we name the inputs as $i_{0,\ldots,63}$ and the check bits as $c_{0,\ldots,19}$ the coding proceeds as follows. The check bits c_0, c_4, c_8, c_{12} and c_{16} are calculated from inputs $i_0, i_4, i_8, \ldots, i_{60}$, the check bits c_1, c_5, c_9, c_{13} and c_{17} are calculated from inputs $i_1, i_5, i_9, \ldots, i_{61}$, and the other two interleaving sections correspondingly. Consequently, the system is capable of detecting error bursts affecting up to 8 adjacent wires and at least two simultaneous single faults extending to 8 simultaneous single faults if only two of them affect the same interleaving section.

In the spare wire approach, four spare wires are added to the system, one for each interleaving section. This gives the system tolerance for permanent error bursts affecting up to 4 wires and maximum of 4 single faults if they affect separate interleaving sections. Thus, the total number of wires is 88, from which 64 are data, 20 check bits and 4 spares.

In the split transmission approach the data is split into two parts, two interleaving sections to each. The data in both parts is doubled, preserving the interleaving. Therefore, the error detection capability is 4-bit wide error bursts and two single errors. The minimum tolerance against permanent errors is determined by the wires located physically in the middle of the link, where the two doubled parts have the minimum physical distance to each others. Since there are four control signals between the parts (see end of the section), the minimum permanent error burst tolerance is 6 bits extending up to 36 bits depending on which part of the link the errors affect. The minimum permanent single error tolerance is one, but a system having even 36 single errors works, if the errors occur only in two different interleaving sections.

The timing of the data transfer between transmitter and receiver is realized using two-phase asynchronous bundled data signaling [10, 18], illustrated in Figure 2(a), while internally the transmitter and receiver use 4-phase signaling [10, 19]. Two-phase signaling is chosen in order to minimize the control wire switching activity and the handshaking delay. This is since in the 4-phase protocol four transitions on handshake lines, two on both request and acknowledge wires, are required, while in the 2-phase protocol only two transitions, one on both request and acknowledge wires, are required. Hence, from this perspective 2-phase signaling protocol is a more attractive choice for NoC interconnects, which could possibly have significant physical wire lengths with high capacitive and resistive properties [20] causing considerable signal delays.

Therefore, in addition to the data wires we need request (*req*) and acknowledgment (*ack*) signals to implement the 2-phase signaling protocol between the transmitter and receiver. For signaling the incorrectness of a data transfer from the receiver to transmitter we use a negative acknowledgment (*nack*) signal instead of *ack*. This makes the backward signaling delay-insensitive since there is no need for making timing assumptions. Finally, we need also a signal for indicating the reconfiguration (*reconf*). The actual reconfiguration data in the spare wire approach can be sent from receiver to transmitter serially by using self-timed dual-rail protocol presented in Figure 2(b) [10], where *ack*/*nack* signals are used for data (0/1) and *req* for acknowledgement. This way also the reconfiguration data exchange is delay-insensitive and no additional signals are needed.

The timing signals are crucial for the correct operation of the link and therefore, they have to be protected against errors. For this purpose we use triple modular redundancy (TMR) as proposed in [21]. Furthermore, the three instances of each control signal are physically dispersed to maintain the tolerance against burst errors. The proposed link structure with spare wires and reconfiguration circuits is presented in Figure 1.

4. REALIZATIONS

The links were designed using the *Haste* design language and *timeless design environment (TiDE)* toolset for asynchronous design by Handshake Solutions [22]. *Haste* has support only for 4-phase bundled data signaling so converter circuits were created to transform signaling to 2-phase and vice versa. The converters were designed using standard components and translated to structural VHDL as were also the majority voters needed for TMR.

To find out the area overhead, performance penalties and the impact on power consumption the introduced fault tolerance causes, also a link without any fault tolerance and a design with ARQ but no reconfiguration circuitry were realized. The buffering capacity of these reference designs was set to the same as in the reconfiguration cases.

4.1. Link with spare wires

The structure of the link with spare wires is presented in Figure 3. The system is pipelined to increase the throughput. The transmitter contains first a latch stage (*L1*), which is

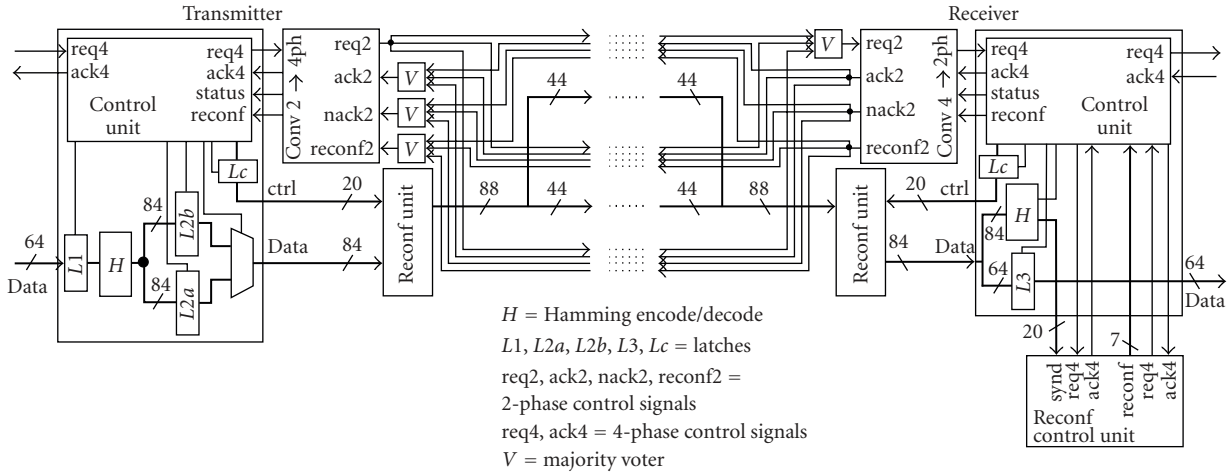


Figure 3: Structure of the designed link with spare wires.

followed by the Hamming encoder circuitry (H). After the encoding the data with check bits is stored in one of the two parallel output latches (*L2a* and *L2b*). These latches are used so that while one is connected to the output channel the next data word can be stored to the other. When *ack* is received to indicate a correct transmission, the output latch can be rapidly changed and a new transmission can begin. In the case of negative acknowledgment (*nack*) the same latch stays connected to the output channel and a retransmission is carried out (see also Figure 2(a)).

In the receiver the error syndrome for incoming data is calculated and the data word is stored to a latch (*L3*). If the syndrome equals zero, *ack* is sent and the data from the latch is forwarded to the receiver output. In the case of a nonzero syndrome, *nack* is sent and the syndrome is passed to the reconfiguration control unit, where it is stored into a 3-place ring buffer, so that the last three nonzero syndromes are found in this buffer. The structure and operation of the reconfiguration control unit is explained in Section 4.3.

When the receiver gets a request together with the reconfiguration vector from the reconfiguration control unit, it switches to the reconfiguration mode. An arbiter is used to guarantee that the mode change cannot occur in the middle of a receive operation. The reconfiguration data exchange between the receiver and the transmitter is illustrated in Figure 4. The receiver sends *reconf* to indicate the mode change to the transmitter. The transmitter acknowledges this through the *req* line. Next, the reconfiguration information (reconfiguration vector, 7 bits) is transferred bitwise using *ack* and *nack* lines and every bit is acknowledged with the *req* line (see Figure 2(b)). After the acknowledgement of the last bit, the receiver sends *reconf* to indicate the mode change back to normal and the transmitter acknowledges it with *req*. Finally, receiver sends *nack*, so that the transmitter sends again the data it was sending when the mode change took place.

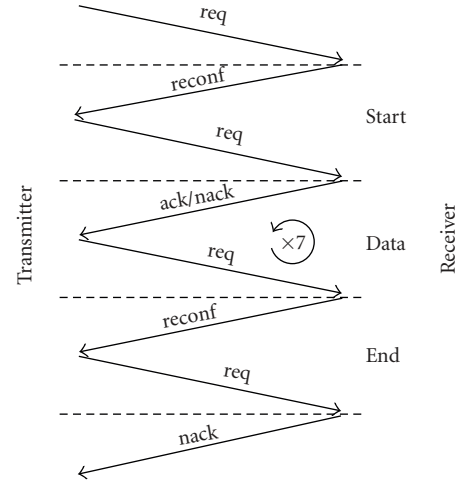


Figure 4: The protocol for sending the reconfiguration information from the receiver to transmitter.

After the transmission of the control word both the receiver and transmitter store the error location into the control register of the correct interleaving section according to the reconfiguration vector. The reconfiguration units were created using structural VHDL. The transmitter side unit is illustrated in Figure 5. Tristate buffers are used to drive the link wires. From the control registers the exact error location is decoded using a 5-to-21 decoder. Based on that value and the fact that no reconfiguration is done above the error location and that all wires below the error location need to be reconfigured, the control signals for the tristate buffers can be solved. By using tristate buffers instead of multiplexers the corrupted wire can be completely isolated from the driving circuit and so, for example, possible short-cuts to power lines do not cause any power leakage. Furthermore, tristate buffers can be easily included in the driving buffers, that are

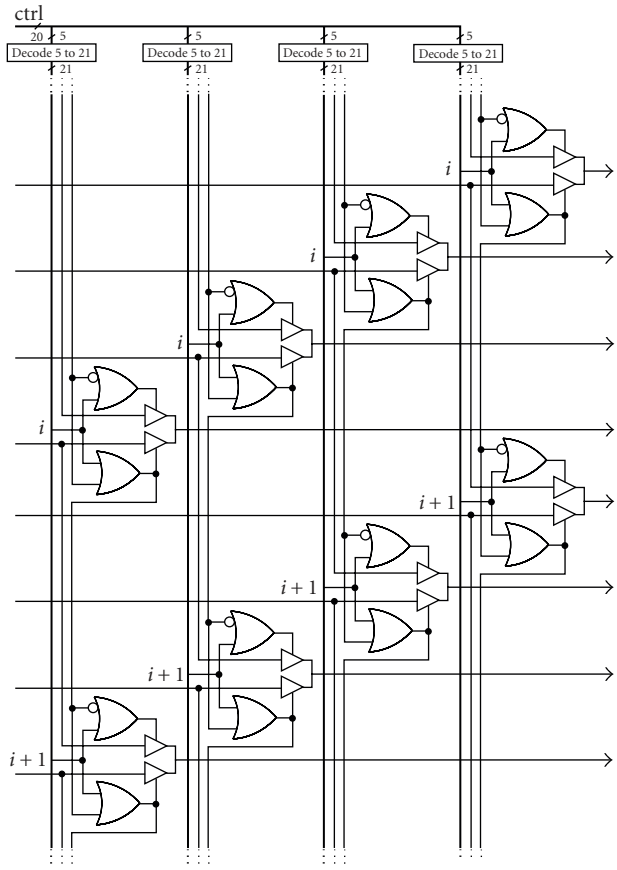


Figure 5: The structure of the reconfiguration unit at the transmitter side.

needed to drive the capacitive on-chip wires. At the receiver side the control is realized similarly with the exception that now multiplexers are used to select the correct wire for each input.

4.2. Link with split transmission

The structure of the link with split transmission is presented in Figures 6 and 7. The system has many similar parts to the one with spare wires. The transmitter differs after the multiplexer *M1*. In the normal mode the output of *M1* is connected to the output of the transmitter via the multiplexer *M3*. In the split mode, indicating that there is a faulty wire in the link, the output of the transmitter is obtained from the multiplexer *M2* (via *M3*). In the Hamming encoder circuitry the data is split into four interleaving sections (every fourth wire per section) and the five check bits are calculated for each section separately and interleaved. In selector *Sel1* either first two or last two sections are chosen and doubled. Splitting the data according to the interleaving sections removes the need of two separate encoding and decoding units. The multiplexer *M2* chooses between the first and second split transmission parts. The part is changed when *ack* is received, in case of *nack* it stays the same and a retransmission is carried out.

In the receiver the error syndrome for incoming data is calculated and the data word is stored into a latch (*L3*) similarly as in the structure with spare wires. The data and check bits are in different parts of the data word depending on the operation mode. This selection is not shown in Figure 6. If the transmission is correct, *ack* is sent to the transmitter and the data from the latch is forwarded to the receiver output via the multiplexer *M5* in case of the normal mode and to the selector *Sel2* in the split mode. The correctness of the transmission is indicated in the normal mode when the syndrome equals zero and in the split mode when either copy of an interleaving section gives a zero-syndrome for both transmitted sections. This means that if the transmitted interleaving sections are marked as 1 and 2 and doubling creates instances *a* and *b*, the transmission is correct if one of the following combinations have zero-syndromes: 1*a* and 2*a*, 1*a* and 2*b*, 1*b* and 2*a*, or 1*b* and 2*b*. In the case of an incorrect transmission, *nack* is sent and the syndrome is passed to the reconfiguration control unit. The syndrome is passed to the reconfiguration control unit also during correct transmissions in the split mode. Based on the syndromes the reconfiguration control unit detects permanent errors and controls the return to the normal mode if the errors have vanished (e.g., they were intermittent). The structure of the reconfiguration control unit and its operation is explained in Section 4.3.

In the split mode, the selector *Sel2* separates the interleaved sections and using the multiplexers *M4a* and *M4b* the correct instances of both transmitted sections are chosen. The control for these multiplexers is obtained from the syndromes calculated by the decoding unit. The first split transmission part is stored to the latch *L4* and when the second part arrives they are interleaved (*Ilv*) and passed to the output via *M5*.

If the receiver is in the normal mode and an error is detected (signal *error* from the reconfiguration control unit) it switches to the split mode. The mode change takes place in the same way in the split mode if the signal *zero* is detected. An arbiter is used to guarantee that the mode change cannot occur in the middle of a receive operation or before a split transmission is completed. The receiver sends *reconf* to indicate the mode change to the transmitter. The transmitter acknowledges this through the *req* line, after which the receiver sends *nack*, in order to get the transmitter to resend the data it was sending when the mode change took place.

4.3. Reconfiguration control

The reconfiguration control unit for the spare wire design is depicted in Figure 8(a) and the unit for the split transmission design in Figure 8(b). Both reconfiguration units have a 3-place syndrome buffer which is divided into four 5-bit segments, one segment for each interleaving section (*LXa*, *LXb* and *LXc*, where *X* is the number of the segment).

In the reconfiguration control unit for the spare wire design, the values of the three syndromes are compared individually in all the four segments and if they equal and are

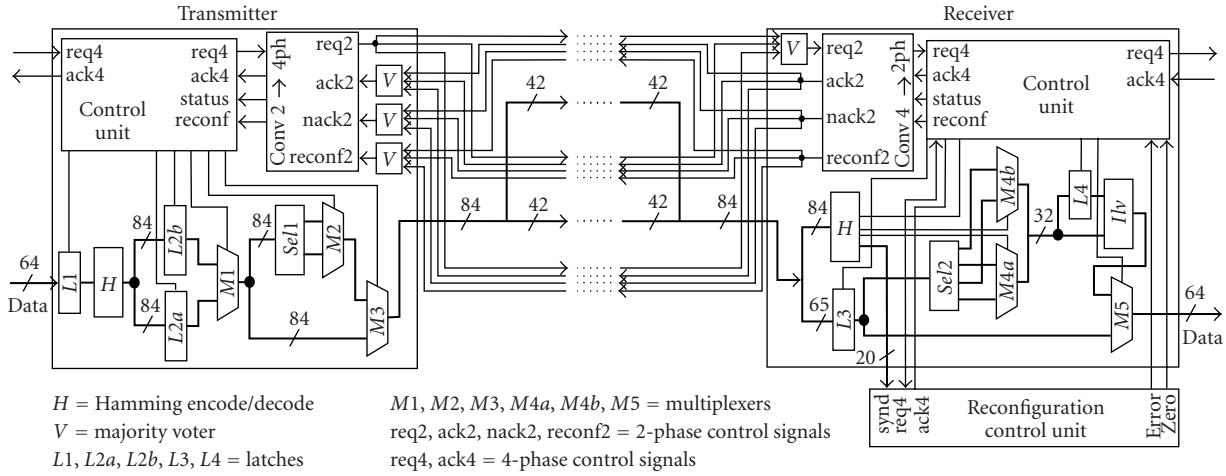


Figure 6: Structure of the designed link with split transmission. Select and interleaver units (*Sel1*, *Sel2*, and *Ilv*) are presented in Figure 7.

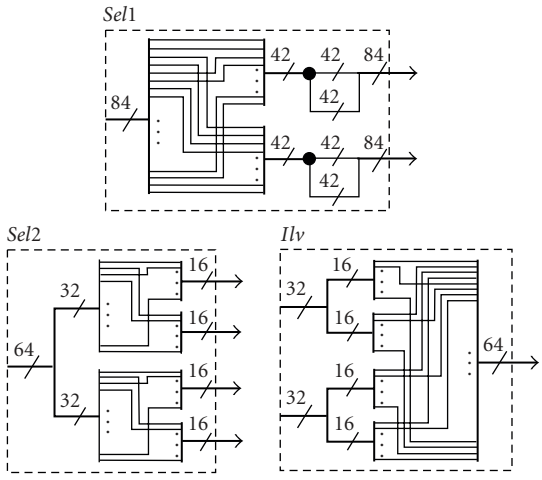


Figure 7: Select and interleaver units.

nonzero, a signal indicating a permanent error in that interleaving section is asserted. Based on these signals the reconfiguration process is commenced. Since there are only one spare wire for each interleaving section, a reconfiguration status flag (latch *Lf*) indicates if the spare has already been used and therefore the reconfiguration cannot take place. Also arbitration between the four error signals is needed to handle the situation, where a permanent error is detected in many sections at the same time (e.g., a burst error).

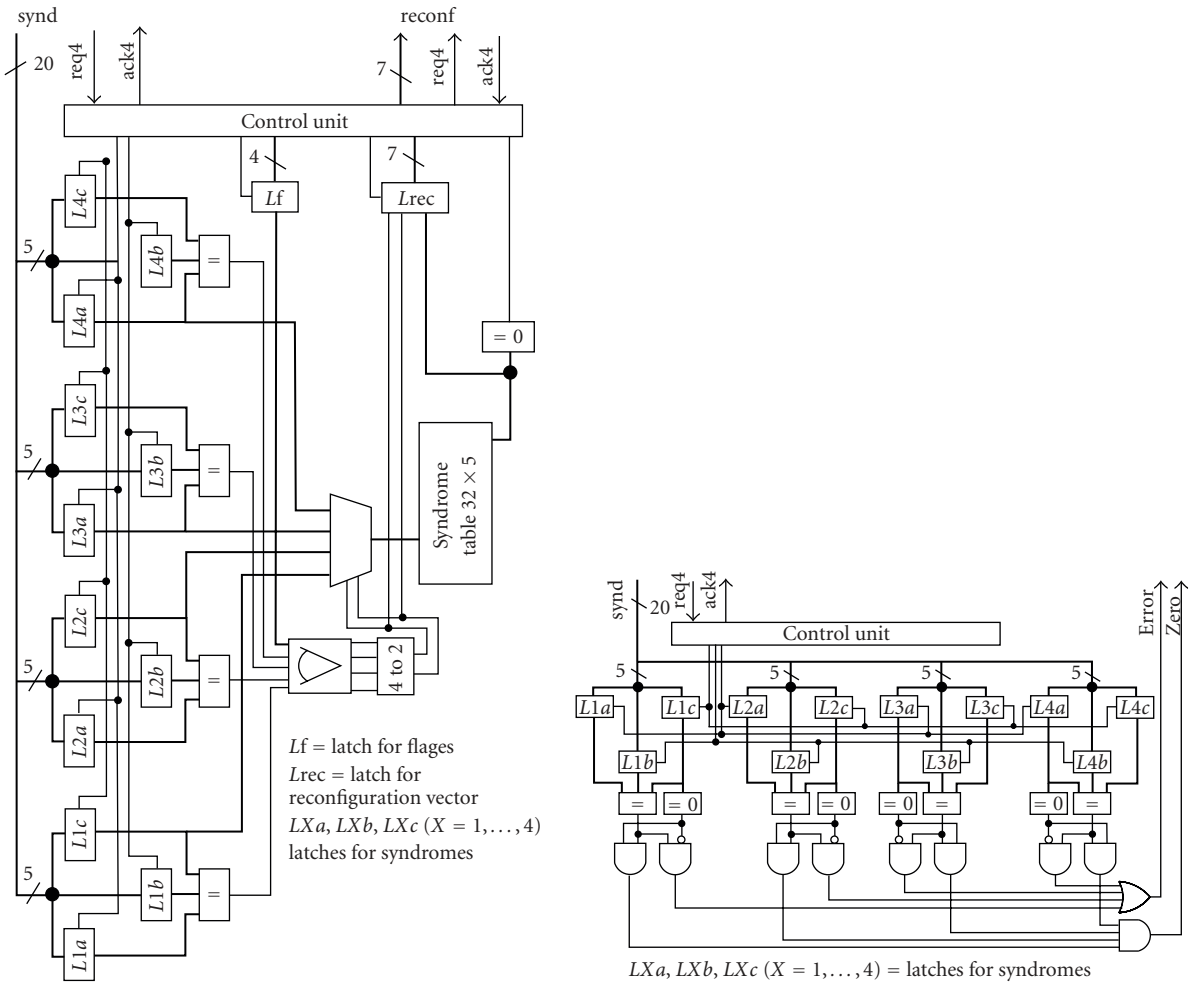
The reconfiguration procedure creates a reconfiguration control vector (latch *Lrec*) to be forwarded to the receiver. This contains 2-bit information indicating the correct interleaving section and 5 bits for pointing out the location of the corrupted wire. The location is fetched from a ROM memory according to the syndrome. In this memory, there is a place for each of the 32 combinations of the 5-bit syndrome. Only 21 of these indicate a single error. The others are set to zero and the circuit checks if the fetched location equals zero. In this case no reconfiguration is done, because the exact error location cannot be solved.

The structure of the reconfiguration control unit for the split transmission design is simpler than the one explained above. An error is indicated via the signal *error* if in one of the segments the three syndromes are equal and nonzero. The signal *zero* on the other hand is asserted if all the syndromes in all the segments are zero.

4.4. Protocol converters

Protocol converters from 2-phase to 4-phase and vice versa in both spare wire and split transmission implementations are shown in Figure 9 [6]. As mentioned earlier, 4-phase signaling [10, 19] is used internally in the transmitter and receiver and in communication between a phase converter and the transmitter/receiver. However, communication between converters, actual communication between the transmitter and receiver, is realized by using a 2-phase handshake protocol to minimize the number of transitions in the link control wires [10, 18]. Hence, protocol converters are required at the boundary of the transmitter and receiver as shown in Figures 3 and 6. Both spare wire and split transmission implementations can utilize the same 2-to-4 and 4-to-2 phase converters.

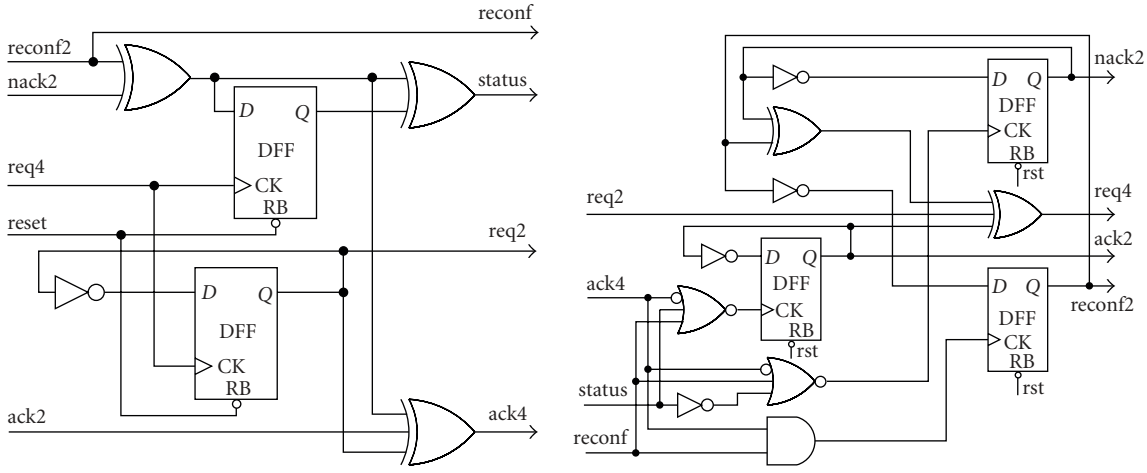
The 4-to-2 phase converter presented in Figure 9(a) is activated by the signal *req4* from the transmitter indicating that a new transaction is going to be commenced. The signal *req4* is connected to the clock input of both flipflops which triggers events in *req2* and *ack4*. The signal *req2* is the actual 2-phase request signal to the receiver indicating availability of new data while the 4-phase signal *ack4* indicates to the transmitter that the data transaction has been started. Immediately after the transmitter has received *ack4* it sets *req4* back to its original level. Observe that the direction of the event in the 2-phase *req2* signal, the actual logic signal level, is not important. Only the change of the signal level is monitored. After the receiver has captured the data it produced an event in the *ack2* signal, which causes *ack4* to go back to its initial



(a) Spare wire design

(b) Split transmission design

Figure 8: Structures of the reconfiguration control units.



(a) 4-to-2 phase converter

(b) 2-to-4 phase converter

Figure 9: Protocol converters.

Table 1: Comparison of the link designs, with wire delay of 500 picoseconds.

Design	No FT	ARQ	Spare	Split
Latency (empty) (ns)	2.67	7.71	8.48	9.65
Latency (full) (ns)	4.93	14.98	17.82	20.31
Throughput (MWord/s)	492.6	204.5	180.5	152.7
Area ($(\mu m)^2$)	6675	12 386	25 377	21 602
Transmitter	4975	8348	12 986	10 518
Receiver	1700	4038	12 391	11 083

Table 2: The burst error probabilities used in simulations.

	Deviation	p_x	Burst length	Explanation
p_0	$\pm[0, \sigma)$	68.3%	1	Single error
p_1	$\pm[\sigma, 2\sigma)$	27.2%	3	+1 adjacent wire in both directions
p_2	$\pm[2\sigma, 3\sigma)$	4.3%	5	+2 adjacent wires in both directions
p_3	$\pm[3\sigma, \infty)$	0.3%	7	+3 adjacent wires in both directions

value informing the transmitter that a new data transaction can be started.

In the case that a retransmission is required, the receiver produces a negative acknowledgment by asserting the signal *nack2*. This changes the state of *status* signal, indicating to the transmitter that the same data should be resent. Events in the signals *ack2* and *nack2* are mutually exclusive, that is, there can be an event only in one of these signals as shown in Figure 2. The signal *reconf2* indicates that the system enters the reconfiguration mode and at the same time causes *ack4* to return to its initial state.

At the receiver side, the 2-to-4 phase converter shown in Figure 9(b), the operation is activated by an event in *req2*, indicating arrival of data. This initiates 4-phase handshaking between the converter and receiver via *req4*. After the receiver has captured data it sets *ack4*, which is connected to the clock inputs of the flipflops through logic gates. At this point, there are three possibilities: *ack2* is sent in the case that data has been properly received, *nack2* is sent if retransmission is required (indicated by *status*), and if reconfiguration is going to take place *reconf2* is sent (indicated by the *reconf* signal) to the transmitter. Regardless of the information sent back to the transmitter, *req4* is initialized back to its initial value which eventually resets *ack4* and hence completes the 4-phase handshake cycle.

5. RESULTS

The designs were mapped to a 130 nm technology, synthesized and simulated using specific VHDL testbenches to find out the best and worst case throughputs and latencies. The interconnect delay was set to 500 picoseconds, which was estimated to model the delay between two routers in an NoC structure including the drivers and possible repeaters. The same wire model was used in all simulations.

The throughputs, latencies and circuit areas are presented in Table 1, where *no FT* means the reference circuit without any fault tolerance structure, *ARQ* the one with ARQ but no reconfiguration properties, *spare* the presented structure with spare wires, and *split* is the structure with the split transmission properties. Latency (full) means the situation, where the buffer capacity of the link is totally occupied when the transmission begins and correspondingly latency (empty) means that there is no data in the buffers. The values listed in Table 1 for the split transmission structure are measured in the normal operation mode. The split transmission latency (full) was measured to be 59.34 nanoseconds and throughput 61.2 Mword/s. The reconfiguration procedure in spare wire design was measured to take in total 29.7 nanoseconds.

6. FAULT TOLERANCE ANALYSIS

In this section the fault tolerance abilities of the presented link structures are analyzed and compared against the reference designs. The used fault model is extended from the one presented in [23] to consider also burst errors as well as intermittent and permanent errors.

According to the model presented in [23] the probability of error ϵ is given by

$$\epsilon = Q\left(\frac{V_{dd}}{2\sigma_N}\right), \tag{1}$$

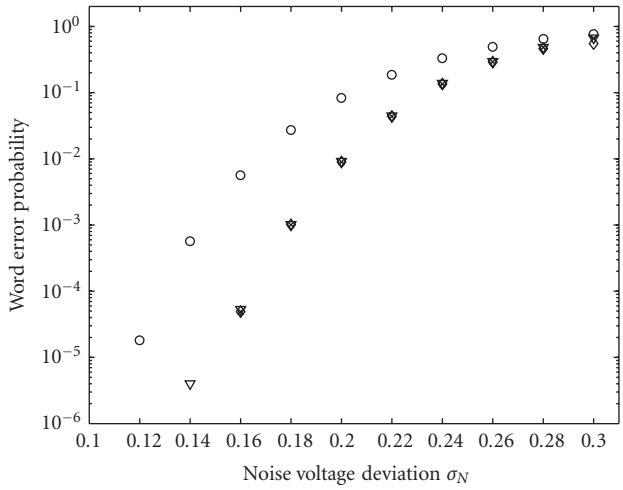
where V_{dd} is the supply voltage, σ_N the deviation of the noise voltage, which is presumed to have normal distribution, and $Q(x)$ is the Gaussian pulse defined as

$$Q(x) = \int_x^\infty \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy. \tag{2}$$

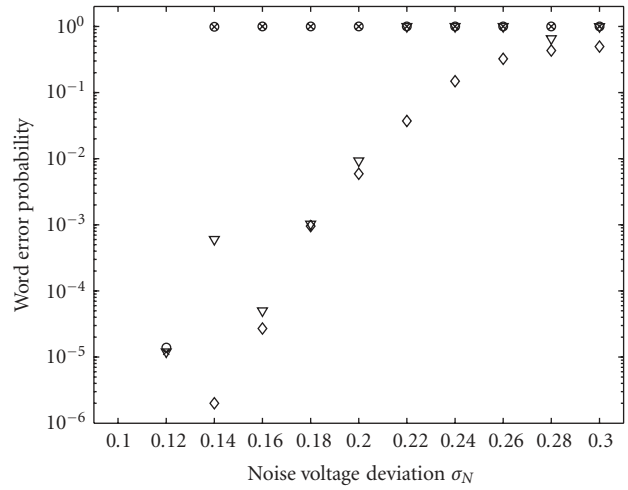
The model assumes that a fault in a single wire is independent of the others. We extend the model by assuming that a fault affects its neighbouring wires in both directions with a certain probability p_1, two neighboring wires with probability p_2 and so on. The probabilities used in the simulations are obtained from the normal distribution in a way that the probability p_0 for a single error is the area under the normal distribution curve for deviations smaller than $\pm\sigma$, p_1 for deviations $\pm[\sigma, 2\sigma)$, and so forth. The probabilities used in the analysis are presented in Table 2.

We add to the model also intermittent and permanent errors. According to [4] 80% of all the errors are transients. We add intermittent and permanent errors respecting this relation and taking the view that half of the added faults are intermittent lasting 10 transmissions and the other half permanent.

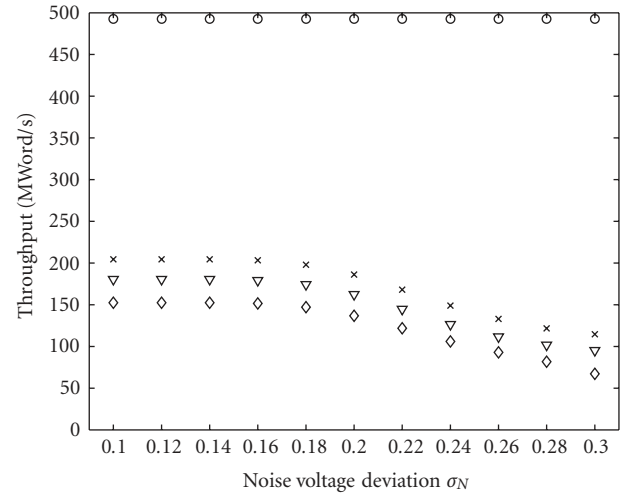
A number of test sets were generated for different values of σ_N while the V_{dd} was held at the constant value 1.2 V. Error sets both with and without intermittent and permanent errors were created. A test data set of one million random data words was generated and simulations were run on the two presented link structures as well as on the two reference structures for the different error sets. The simulation results are shown in Figure 10.



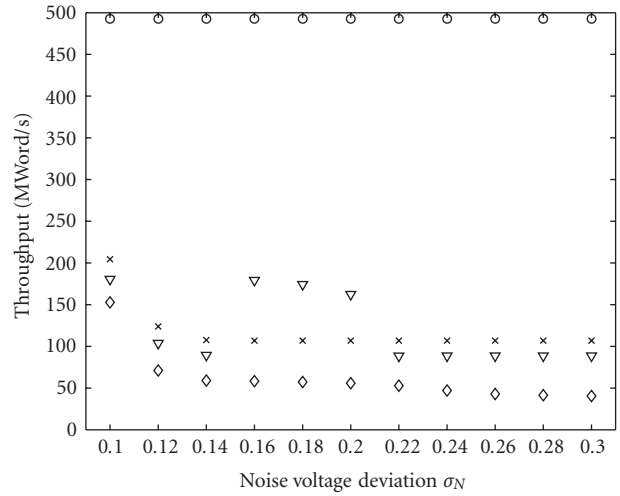
(a) Error probability in the case of only transient errors



(b) Error probability in the case of transient, intermittent, and permanent errors



(c) Throughput in the case of only transient errors



(d) Throughput in the case of transient, intermittent, and permanent errors

Figure 10: The performance of different structures in the presence of different error scenarios.

The word error probability as a function of the noise voltage deviation is shown in Figure 10(a) for the error model without intermittent and transient errors. The fault tolerance properties of the two presented structures match with the reference circuit with ARQ but no reconfiguration. This is quite obvious since the fault tolerance method they use is the same. Quite obvious is also the fact, that the structure without any fault tolerance properties results in a higher word error probability rate as the others.

The corresponding throughputs are shown in Figure 10(c). We see that the throughput decreases as the noise deviation increases for all the other structures except for the one without fault tolerance properties, which have constant throughput regardless of the amount of faults. The ARQ is realized in a way that in the case of a detected fault, the word is retransmitted once, but no more. If there was no limit for the number of retransmissions the circuit would end up in a livelock. The effect of most transient faults, for example, intersymbol interference, radiation induced pulses, crosstalk, and so forth, ceases by just waiting before sampling the values on the line. For this reason the number of retransmissions is limited to one. Unfortunately, our error model does

Table 3: Energy consumption analysis of the link designs.

(a) Energy per transmission for $\sigma = 0.10$

Design	No FT	ARQ	Spare	Split
Transmitter (pJ)	26.28	28.31	29.50	30.94
Receiver (pJ)	7.12	12.42	14.46	17.75
Total (pJ)	33.41	40.74	43.96	48.69
Compared to ARQ			+7.9%	+19.5%

(b) Energy per transmission for $\sigma = 0.20$

Design	No FT	ARQ	Spare	Split
Transmitter (pJ)	26.28	28.98	29.57	34.25
Receiver (pJ)	7.80	21.24	16.30	46.37
Total (pJ)	34.08	50.22	45.86	80.62
Compared to ARQ			−8.7%	+60.5%

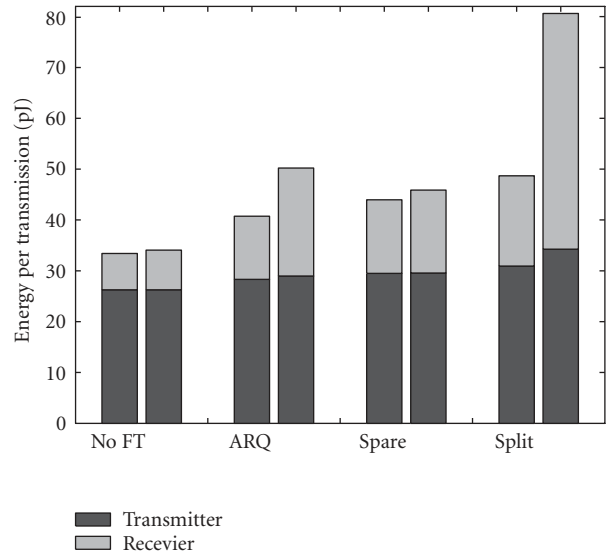


Figure 11: Energy consumption analysis of the link designs.

not take into account this phenomenon but instead all transfers are presumed to be independent of each others. This gives pessimistic values for the fault tolerance of the simulated structures. Nevertheless, they can be compared with the used model since the situation is the same for all of them. The impact of the limited number of retransmissions is seen in the saturation of the throughputs as the noise voltage deviation is increased. The throughput-based ranking order of the structures corresponds to the values presented in Table 1 and it remains the same for all the simulated sets.

The simulation results for the error sets with also intermittent and permanent errors are shown in Figures 10(b) and 10(d). The reference structures lose their operability rapidly as the noise voltage deviation is increased. As expected, they do not work in the presence of permanent faults. The presented online reconfigurable structures maintain their operability although there is more variance in the results. The variance is originated from the randomness of the error locations and the properties of the structures to guarantee tolerance against some limited set of permanent faults while a larger set is tolerated with some conditions. From the two presented structures the one with split transmission shows higher fault tolerance.

The throughputs presented in Figure 10(d) show similar changes as in the simulations without intermittent and permanent faults. For the reference structure without fault tolerance, the throughput is constant and for the reference structure with ARQ the throughput decreases, now more rapidly than without intermittent and permanent faults, and saturates because of the limited number of retransmissions. The spare wire approach has exceptions to this trend. When the reconfiguration has been successfully carried out, the permanently faulty wires have been replaced by spares, the throughput equals the fault-free situation and the word error probability remains low. On the other hand, if the reconfiguration is not successful the fault tolerance is almost as low as for the reference circuit with ARQ, and the throughput has similar values as in the case without intermittent and permanent faults for high noise voltage deviation. The split transmission structure shows a similar curve to the one in the case without intermittent and permanent faults, but with intermittent and permanent faults it saturates to a lower throughput value. This is because the throughput in the split mode is lower and in the presence of a permanent fault, the circuit remains in the split transmission mode all the time.

7. POWER ANALYSIS

The power consumption of the systems was analyzed by using Synopsys Primepower. Simulations were carried out for an input set of ten thousand random words for two different error sets containing transient, intermittent, and permanent errors. The error sets with noise voltage deviations $\sigma = 0.10$ and $\sigma = 0.20$ were chosen. The first one demonstrates a situation of low error probability whereas the second contains a lot of errors. By comparing these two analyses the power efficiency in the presence of errors can be seen. The simulation results are presented in Table 3 and in Figure 11, where *no FT* means the reference circuit without any fault tolerance structures, *ARQ* the one with ARQ but no reconfiguration properties, *spare* the presented structure with spare wires, and *split* is the structure with the split transmission properties. The average power consumption is translated to energy consumption per transmitted word by taking into consideration the throughputs of the different systems and averaging over the random input set.

The energy consumption values do not include the energy taken by the drivers driving the link wires. These values were not included since the main target of the analysis is to see the impact of the added reconfiguration properties in comparison to the reference circuit with ARQ but no reconfiguration. These values are comparable since in all of

these systems the number of active wires is the same or the difference in the wire count can be ignored. In the spare wire approach, the added spare wires do not consume any power and therefore, the number of active data wires is 84 as is the case in the split and reference designs. In the systems with reconfiguration properties, there is one additional tripled control signal (*reconf2*), but it is switched extremely rarely. The power analysis of the majority voters indicates that the power consumption of the majority voters connected to the *reconf2* signals is less than one thousandth of the corresponding values of the ones connected to the *ack*2 signals. The relative differences of the energy consumptions of the spare and split approaches as compared to the reference with only ARQ are presented in Table 3 for the both simulated noise voltage deviations.

The results indicate that the addition of reconfiguration properties results in a small increase in power consumption. For the spare wire approach the increase is less than 8% in the case of few errors, as for the split transmission design the increase is almost 20%. When the noise voltage deviation is increased the values show dramatical but nevertheless expectable results. Now the spare wire approach gives the best power consumption values, almost 9% less than for the reference design. The reasons for this were already discussed in the previous section. The power consumption of the spare wire approach is almost the same as without any permanent or intermittent errors when the reconfiguration is successful. The split transmission design uses a lot more energy than the others, which is rather obvious since it takes many transmission for all the words.

From the power consumption simulations it can be concluded that the spare wire approach is more power efficient than the split transmission approach, and that the addition of spare wires and the needed reconfiguration properties does not cause any considerable changes in the power consumption.

On the other hand, the results also indicate that the addition of fault tolerance properties in general results in a higher power consumption. This observation is further strengthened by the fact that the system without fault tolerance properties contains less wires than the others.

8. DISCUSSION

When comparing the designs with the reconfiguration properties to the ARQ design without them, it can be noticed that there is a performance penalty. In the spare wire design, the latency increases 10%/19% (empty/full) and the throughput decreases 12%, while in the split transmission design the latency increase is 25%/36% and the throughput decrease 25%. Thus, the spare wire approach has a smaller impact on the performance. On the other hand, the area overhead is larger in the spare wire approach. It uses 105% more area than the ARQ design without reconfiguration. The corresponding value for the split transmission is 75%. The area increase is mainly due to the reconfiguration control unit in the receiver. Moreover, the reconfiguration logic in both the transmitter and receiver consume area.

The split transmission performance should be compared to the situation, where the erroneous link is bypassed through neighbouring links and routers in an NoC. The split transmission latency (full) is 14.4 nanoseconds larger than three times the latency of the reference circuit without split transmission properties. The three link latencies corresponds to the shortest route to bypass a link in a mesh-shaped NoC. In addition to the link latencies, the latencies of the two routers contribute to the total delay of the bypass route. The latency of these two routers is comparable to the 14.4 nanoseconds. Therefore, the latency of the split transmission is approximately the same as that of bypassing the erroneous link. The essential difference is that bypassing increases traffic in neighbouring links and may cause serious consequences in the form of congestion. In the worst case this affects the operation of the whole NoC.

The area overhead should be considered in the NoC context. Let us consider a simple mesh structure, where one router is assigned for each resource and the size of a resource is two times two millimeters [24]. There are three bidirectional links per each resource, thus, in total six transmitter/receiver pairs per resource. The area overhead caused by the introduced fault tolerance is only 2.2%–2.8% compared to the situation without any fault tolerance. The router area was not included in this calculation but it further decreases the relative area overhead.

If the link is connected, for example, between NoC routers, the buffering capacity of the link can be taken into account, and so the buffering capacity of the routers can be decreased by the same amount. This obviously reduces the area overhead. In the same way the tristate buffers of the transmitter reconfiguration unit in the spare wire approach can be a part of the link drivers as was already discussed in Section 4.1.

The simulations were carried out using a 130 nm technology. As the dimensions have been scaled below 100 nm the delay of wires has not decreased as is the case with the delay of logic. The gap between the wire and logic delays is expected to increase as scaling deeper into nano regime [4]. Based on this scenario and because the bottleneck at the moment are the receiver and transmitter units, the performance of the presented link structures should increase as the dimensions are scaled down.

The main parts were created using a modelling language and synthesis tools which use a rather conservative approach to the problem. Based on our previous experience we predict that the performance could be significantly enhanced by careful manual design although still using standard gate libraries [6].

In the presented designs the approach for transient fault tolerance was the use of error detection and ARQ with stop-and-wait strategy. The addition of reconfiguration to tackle intermittent and permanent errors does not force sticking at this approach. Quite as well forward error correction and in the spare wire approach also different ARQ strategies [15] could be used. One such strategy is go-back-N, where instead of waiting for an acknowledgment of correct transfer, the transmission proceeds with next word instanteously and in

the case of an error the system returns the necessary amount of words (N) backwards. Another strategy is selective repeat, where the transmission is continued as in go-back-N, but in the presence of an error only the erroneous word is asked to be retransmitted. In the presented analysis the focus has been on the comparison of the reconfigurable structures with the reference structure with ARQ but no reconfiguration. Since all these designs use the same ARQ strategy, the choice of it has not been of vital importance and hence, the simplest one has been used.

The reconfiguration control indicates a permanent error when the same error syndrome has been repeated three times. In the split transmission approach the reconfiguration could be advantageous also under other conditions. In Figure 10(a), we see a slightly smaller error probability for split transmission than for the other designs when $\sigma_N = 0.30$. In this case the circuit has entered the split transmission mode since the temporary error probability has been so large that the same error syndrome has been repeated three times. The lower throughput for this situation confirms the diagnosis. This gives an idea that the change to the split transmission mode might be advantageous if the error occurrence rate crosses some limit.

The tolerance against single permanent faults in the split transmission design is limited by the fixed location of the doubled data parts. If the location of the permanent fault was more exactly isolated, the placement of the doubled data could be controlled and a greatly improved tolerance against single faults could be achieved. The error location could be determined using the syndromes as is done in the spare wire design. This would probably increase the control logic and would also result in performance degradation. The presented approach has the assumption that if there are multiple errors, they most likely occur as bursts, which can be motivated by considering different manufacture defects.

The split transmission design has a property to return to the normal mode if the fault was intermittent. A similar property would be advantageous also to the spare wire design, where the reconfiguration at the moment is irreversible. The circuit is not able to tolerate permanent errors if there have been intermittent errors in the same interleaving section (but at a different wire) although the intermittent fault has vanished. The intermittent faults in practice repeat themselves in the same locations due to, for example, small anomalies in the manufactured chip, which turn into faults under certain environmental conditions. However, the used fault model considers all the intermittent faults independent of each others and therefore gives pessimistic values for the fault tolerance properties of the spare wire approach. The advanced detection of different fault scenarios will be a part of our future research.

The number of spare wires in the presented design was set to four and they were assigned one for each interleaving section. The fault tolerance properties could be enhanced by increasing the number of spare wires. It would also be advantageous to have the spare wires assigned more flexibly to different interleaving sections. One interleaving section may have many erroneous wires while some other has none. The drawback of the increased flexibility is the growth of complexity which probably leads to a performance decrease and area overhead.

In this work, the transmitter and receiver circuits were presumed error-free. When considering fault tolerance of future nanoscale systems, also these circuits should be taken into consideration from the fault tolerance perspective. We will address this issue as a part of our future work.

9. CONCLUSION

We proposed link structures that have properties for tolerating efficiently transient, intermittent, and permanent errors. The protection against transient errors was realized using Hamming coding and interleaving for error detection and ARQ as the recovery method. Two approaches were introduced to tackle the intermittent and permanent errors. Split transmission was an approach utilizing time redundancy while the other structure, which introduced spare wires, was a hardware redundancy approach. Communication in the links was based on asynchronous 2-phase signaling and the control signals for ARQ and reconfiguration were incorporated into these control signals. The control lines were protected using triple modular redundancy.

We presented designs for the different components of the links and proposed the needed reconfiguration control. The designs were implemented, simulated and compared against reference designs. The simulation results show that the performance decrease when comparing to a design with ARQ but no reconfiguration is larger for the split transmission design (latency 31%/throughput 25%) than for the spare wire design (15%/10%) while the area overhead is larger for the spare wire design (105%) as for the split transmission design (75%). The fault tolerance analysis using error models containing temporary, intermittent, and permanent faults that occur as both bursts and single errors, shows the effectiveness of the presented link structures. When there are no intermittent or permanent errors present, the structures perform the same way as the reference design with ARQ but no reconfiguration, and which clearly outperforms the reference design without any fault tolerance. When also intermittent and permanent errors are taken into account, the presented link structures show clearly better results than the reference designs. From the two presented reconfiguration structures, the split transmission design tolerates faults slightly better than the design with four spare wires. On the other hand, the spare wire approach turned out to be more power efficient than the split transmission design. Using the spare wire design the power consumption is approximately the same as with the reference design with ARQ but no reconfiguration. In the presence of an excessive amount of errors the spare wire approach turns out to be even more energy efficient than the reference design.

Our research shows that combining different fault tolerance methods a structure capable of tolerating all different types of errors is achievable. As is always the case with fault tolerance, this does not come for free. There is a clear area overhead, but on the other hand when comparing to the IP

block areas in the NoC context, the overhead is only a couple of percents.

ACKNOWLEDGMENTS

The authors would like to thank The Finnish Cultural Foundation, The Foundation of Technology, and Ulla Tuominen Foundation for their financial support. They also acknowledge the financial support from Academy of Finland.

REFERENCES

[1] C. Constantinescu, "Trends and challenges in VLSI circuit reliability," *IEEE Micro*, vol. 23, no. 4, pp. 14–19, 2003.

[2] International Technology Roadmap for Semiconductors, 2005, http://public.itrs.net/.

[3] T. Lehtonen, J. Plosila, and J. Isoaho, "On fault tolerance techniques towards nanoscale circuits and systems," Tech. Rep. 708, Turku Centre for Computer Science (TUCS), Turku, Finland, August 2005.

[4] G. De Micheli and L. Benini, *Networks on Chips*, Morgan Kaufmann Publishers, San Francisco, Calif, USA, 2006.

[5] D. M. Chapiro, *Globally-asynchronous locally-synchronous systems*, Ph.D. thesis, Stanford University, Stanford, Calif, USA, October 1984.

[6] P. Liljeberg, J. Plosila, and J. Isoaho, "Self-timed communication platform for implementing high-performance systems-on-chip," *Integration, the VLSI Journal*, vol. 38, no. 1, pp. 43–67, 2004.

[7] J. Muttersbach, T. Villiger, and W. Fichtner, "Practical design of globally-asynchronous locally-synchronous systems," in *Proceedings of the 6th International Symposium on Advanced Research in Asynchronous Circuits and Systems (ASYNC '00)*, pp. 52–59, Eilat, Israel, April 2000.

[8] M. Renaudin, "Asynchronous circuits and systems: a promising design alternative," *Microelectronic Engineering*, vol. 54, no. 1-2, pp. 133–149, 2000.

[9] C. L. Seitz, "System timing," in *Introduction to VLSI Systems*, C. Mead and L. Conway, Eds., chapter 7, Addison-Wesley, Reading, Mass, USA, 1980.

[10] J. Sparsø and S. Furber, *Priciples of Asynchronous Circuit Design—A System Perspective*, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2001.

[11] D. Bertozzi, L. Benini, and G. De Micheli, "Error control schemes for on-chip communication links: the energy-reliability tradeoff," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 24, no. 6, pp. 818–831, 2005.

[12] S. R. Sridhara and N. R. Shanbhag, "Coding for system-on-chip networks: a unified framework," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 13, no. 6, pp. 655–667, 2005.

[13] L. Li, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin, "Adaptive error protection for energy efficiency," in *Proceedings of IEEE/ACM International Conference on Computer-Aided Design (ICCAD '03)*, pp. 2–7, San Jose, Calif, USA, November 2003.

[14] H. Zimmer and A. Jantsch, "A fault model notation and error-control scheme for switch-to-switch buses in a network-on-chip," in *Proceedings of the 1st IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS '03)*, pp. 188–193, Newport Beach, Calif, USA, October 2003.

[15] R. E. Ziemer and R. L. Peterson, *Introduction to Digital Communication*, Prentice-Hall, Upper Saddle River, NJ, USA, 2nd edition, 2001.

[16] B. W. Johnson, *Design and Analysis of Fault-Tolerant Digital Systems*, Addison-Wesley, Reading, Mass, USA, 1989.

[17] C. Grecu, A. Ivanov, R. Saleh, and P. P. Pande, "NoC interconnect yield improvement using crosspoint redundancy," in *Proceedings of the 21st IEEE International Symposium on Defect and Fault Tolerance in VLSI Systems (DFT '06)*, pp. 457–465, Arlington, Va, USA, October 2006.

[18] I. E. Sutherland, "Micropipelines," *Communications of the ACM*, vol. 32, no. 6, pp. 720–738, 1989, The 1988 Turing Award Lecture.

[19] A. Davis and S. M. Nowick, "Asynchronous circuit design: motivation, background and methods," in *Asynchronous Digital Circuit Design*, G. Birtwistle and A. Davis, Eds., pp. 1–49, Springer, New York, NY, USA, 1995.

[20] W. J. Dally and J. W. Poulton, *Digital Systems Engineering*, Cambridge University Press, Cambridge, UK, 1998.

[21] S. Murali, T. Theocharides, N. Vijaykrishnan, M. J. Irwin, L. Benini, and G. De Micheli, "Analysis of error recovery schemes for networks on chips," *IEEE Design & Test of Computers*, vol. 22, no. 5, pp. 434–442, 2005.

[22] Handshake Solutions, http://www.handshakesolutions.com/.

[23] R. Hegde and N. R. Shanbhag, "Toward achieving energy efficiency in presence of deep submicron noise," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 8, no. 4, pp. 379–391, 2000.

[24] W. J. Dally and B. Towles, "Route packets, not wires: on-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, pp. 684–689, Las Vegas, Nev, USA, June 2001.