Phase-Encoding for On-Chip Signalling

Crescenzo Sabato D'Alessandro, *Student Member, IEEE*, Delong Shang, Alexander Bystrov, Alexandre V. (Alex) Yakovlev, *Senior Member, IEEE*, and Oleg Maevsky

***Abstract*—A novel self-timed communication protocol is based on the phase-modulation of a reference signal. The reference signal is sent on a number of transmission lines and the data can be recovered observing the sequence of events on the these lines. Employing several lines increases the number of states hence reducing the number of symbols required for a transmission. A new encoding algorithm is described which generates symbol-dependent matrices which are used to control the phase of transmission lines. The protocol concept, the algorithm and analysis of the system, together with simulation results, are presented.**

***Index Terms*—Digital communication, encoding, fault tolerance, interconnects circuit interconnect.**

I. Introduction

THE issue of fast and reliable communication fabric is crucial for the successful design of systems-on-chip. An approach to design of such a communication fabric is the network-on-chip (NoC) [1]. The synchronization of blocks is a nontrivial aspect of design and research has delivered interfaces which are *self-timed* and *speed-independent* to address this problem. An example of self-timed interface can be found in [2], where the communication between two separate clock domains is investigated. Transient errors due to cross-talk, cross-coupling, ground bounce or environment interference become more prominent as integration increases, and this effect can be observed interconnect wires, as underlined by the work of Dupont and Nicolaidis [3], [4]. Quoting Nicolaidis: "it is predicted that single-event upsets (SEUs) induced by alpha particles and cosmic radiation will become a cause of unacceptable error rates in future very deep submicron and nanometer technologies. This problem, concerning in the past more often parts used in space, will affect future ICs at sea level" [4]. This motivates the fault tolerance approach to design. In [5] a novel approach is proposed where, using a dual-rail scheme, the data is sent using the phase relationship between two differentially delayed copies of a reference signal. A mutual exclusion (ME) element [6] can be employed as a PD for data recovery. In the same paper it is shown that the use of such method introduces a number of interesting characteristics which makes the approach particularly robust to SEUs.

The system resembles a phase-shift-keying (PSK) communication channel. This communication scheme employs a number of phases to encode data; the receiver clock is synchronized to the sender clock and the phase difference between symbols indicates the item of data being transmitted. This scheme is very robust and allows for fast transmission; however, a direct implementation of the scheme for on-chip communication would be prohibitively costly in terms of power and area consumption. The scheme described in [5] uses only two lines and the phase relationship is binary, in the sense that it is either greater or less than zero. In order to obtain the same advantages of multiple phases on-chip, in [7] the use of multiple-rail phase encoding is proposed with a preliminary analysis of fault tolerance, bandwidth and design solutions for this communication scheme. The use of several wires increases the number of states which can be encoded in one symbol; it is shown that the use of phase-encoding over multiple lines increases dramatically the number of bits per symbol, increasing bit-rate without a significant increase in power consumption. Symbols are encoded using a new algorithm which generates unique symbol-dependent matrices. This algorithm is of simple implementation and can be implemented with little logic.

This paper describes the novel concept of phase-encoding communication and highlights some useful characteristics of this scheme: 1) as symbols are recovered only after all lines have switched spurious transitions are filtered out improving robustness; 2) the fact that the spacers alternate increase fault detection; 3) using a number of wires greater than 4 the number of bits per symbol is greater than using a parallel bus; and 4) because all wires switch when a symbol is sent, it is possible to regenerate the symbol along a communication line to counteract signal degradation over long wires. The remainder of the paper is organized as follows. Section II describes the basic phase-encoding concepts; Section III discusses the issues related to the design of repeaters for phase-encoding; Section IV provides an analysis of fault detection. A discussion of link frequency and bitrate achievable with phase-encoding links is provided in Section V; Section VI compares the method with other protocols; an implementation example and simulation results are provided in Section VII; Sections VIII and IX finally conclude the work.

II. Phase-Encoding Concept

Dual-rail codes are designed in such a way that data is sent across two lines rather than a single one. The data is encoded by switching high or low one of the lines; the difference in level represents a symbol. The traditional dual-rail protocol employs a single spacer, whereby after each transmission of a valid bit of data the bus returns to the zero state. In [8] the use of alternating spacers is introduced and several implications with respect to security (power signatures) are analyzed. The paper concludes that the alternating spacer protocol (ASP) is good for security, the circuits implementing it are easy to synthesize in standard

Manuscript received July 14, 2006. This work was supported by EPSRC under Grant EP/C512812/1. This paper was recommended by Associate Editor R. Puri.

C. D'Alessandro, D. Shang, A. Bystrov, and A. V. Yakovlev are with the Microelectronics System Design Group, Newcastle University, Newcastle upon Tyne NE1 7RU, U.K. (e-mail: cdalessandro@ieee.org).

O. Maevsky is with Intel Labs, Moscow 125252, Russia.

Digital Object Identifier 10.1109/TCSI.2008.916407

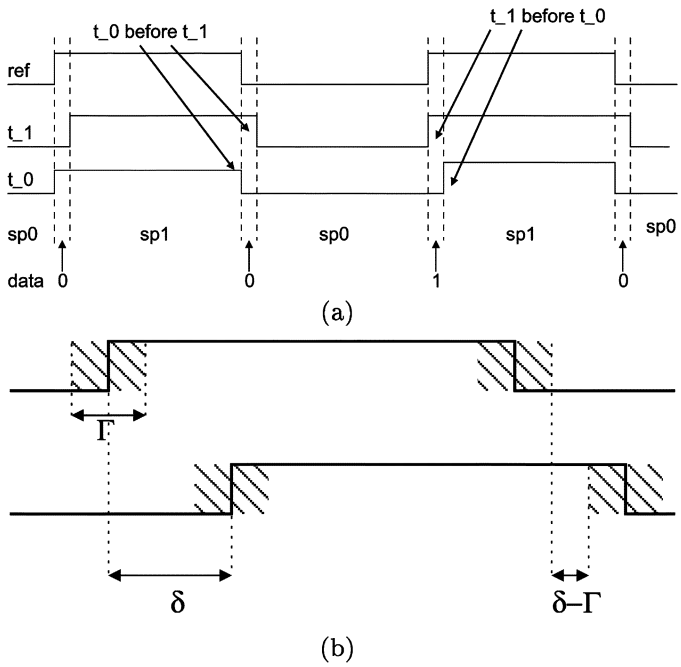

Fig. 1. Example of waveforms for dual-rail phase encoding.

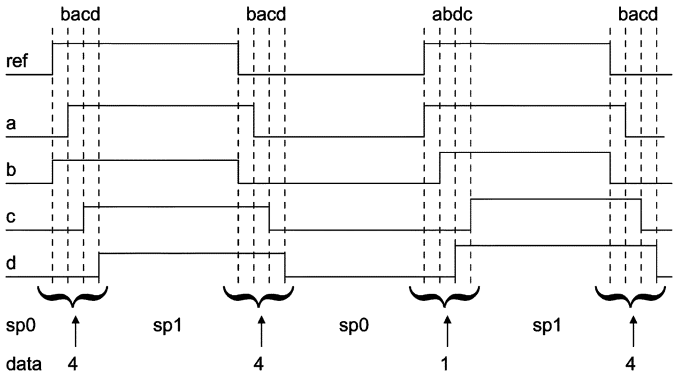

Fig. 2. Example of waveforms for multiple-rail encoding.

gates and the whole approach can be integrated in the standard design flow. The protocol described in the present paper resembles the ASP, while retaining conceptual differences.

For the transmission method to work the sender and the receiver *can* be internally synchronous systems, albeit within uncorrelated clock domains; alternatively they can be fully-asynchronous blocks, or combinations of the two. The reference signal is used for sending data on its rising and falling transitions. The data being sent modulate the phase of the reference differently on each transmission line by controlling the variable delay elements (VDE), as in Fig. 1(a). The receiver recovers the data by comparing the phase of the two signals to each other. Data values are encoded as the sign of the phase difference of the reference signal on the transmission lines. Rather than measuring the phase difference, the receiver decodes the data by observing the sequence of events on the transmission lines. The receiver records the data upon the arrival of the first transition, but the bit validity is recorded *only when the next spacer settles on the transmission lines*. Therefore, the measure of interest is the *differential delay* introduced by the VDEs, indicated here with δ [Fig. 1(b)]. This differential delay introduces an *event window* where an imbalance on the lines is present. The size of this event window is determined by δ, which indicates the "nominal value" for this window, and the jitter introduced by the channel on each line Γ. Provided that the system is able to reject transient faults appearing outside the event window, one such fault will generate an error only if it happens within the window. Effectively, we reduce the event window to a minimum in order to minimize the effect of transient faults, while still recognizing data.

In order for the data to be correctly recovered, δ must be recognized at the receiver (but not measured); in [5], synchronization of the link is discussed and here we limit to remind that to ensure reliability, the jitter Γ introduced by the channel must be taken into account so that $\delta \gg \Gamma$ in order to guarantee deterministic behavior of the system. However, if T is the clock period, $\Gamma \ll T$ and therefore δ can be chosen so that $T > \delta \gg \Gamma$. The value of Γ can be estimated using various techniques [9].

The fact that valid data is recorded only when the next spacer is generated (and therefore when both transmission lines have changed status) has an important property: a hazard on one of the transmission lines (generated by cross-talk, electromagnetic interference, cosmic radiation) will not be recorded as data. This is particularly important if several SEUs can be generated by the environment; provided the events do not affect both lines *at the same instant*, the system will ignore the error.

The concept of phase encoding can be extended to multiple rail communication (Fig. 2). In general, multiple rail encoding is achieved by switching a number n of lines out of a number m of communication rails. The work of Bainbridge, summarized in his thesis [10], describes the use of a single-rail bus to avoid the overhead imposed by the use of multiple-rail bus implementations; however, he also proposes the use of 1-of-4 encoding [11] as a possible improvement in terms of power efficiency.

The phase encoding method proposed in [7] is such that all wires toggle during a symbol transmission and the receiver monitors the sequence of arrival of edges over the lines. Given a number of available lines n, assume that the delays introduced onto the wires are all different, that is, no two wires present the same delay. Then, the order of arrival in the ideal case (no noise on the transmission lines) will correspond to one of $n!$ states, as the possible orders of arrival will correspond to the permutation of n different objects. These states can be therefore transmitted as a single symbol on the wires. If the time difference between each pair of edges is δ the time required to transmit a symbol is $n\delta$; therefore the frequency of the link cannot exceed $1/2n\delta$. This is also the maximum frequency allowed on each line. A more accurate description of link frequency issues is proposed in Section V.

The system relies on the signals seen from the receiver's phase detector (PD) being aligned with respect to their relative phase. This can deteriorate in long wires, as the cross-talk between the two lines is dependent on the coupling capacitance of the wires, which in turns depends on their length. If however the wires are routed differently, increasing the physical distance between them, any mismatch in length and in number of vias

would cause difference in wire resistance; the overall capacitance between the wire and the ground planes can also be affected if the wires "hop" across layers. These two effects result in a mismatch between the time constants of the wires and therefore the required delay relationship will be corrupted.

The effect of cross-talk due to coupling capacitance on a multiple-rail channel has been described in [7]. A method to avoid this problem consists in increasing the physical spacing between the wires: the effects described in [7] referred to minimum-pitch communication lines. Instead, increasing the pitch reduces the effects significantly. As it might be expected, increasing the spacing between wires decreases the maximum corruption almost linearly; increasing the width is less effective. From simulations obtained using Metal5 layer wires in a 0.18-μm process with a π-model the maximum corruption was, in the case of minimum pitch, 64%; this was reduced to 34% if the spacing was doubled and 33% if the width was also doubled. The nominal input time separation δ was 100 ps.

III. Repeaters

This section consists of a review of the material presented in [12], with the aim to describe the issues involved with the successful design of an appropriate repeater for phase encoding. The section does not provide an exhaustive discussion on the design of such devices, rather it proposes a range of implementation solutions and highlights some of the most prominent problems enountered in the design of repeater devices.

As described in the previous section, the system relies on the preservation of a given time difference between two related events on two different communication lines: as described in the previous section the assumption that the event separation is preserved along the path cannot be maintained. However, it is clear that this assumption still holds for a short path, as the cross-talk induced by each line on the neighbouring one is limited, thanks to the limited coupling between the wires. Therefore, a method to preserve the time separation between events is required in order to enable long-distance signalling. We recall that with the word "repeater" we intend a device which, given as input two identical signals delayed by a variable amount outputs the same signals with time delay always $\pm\delta$. In particular, in this work we focus on the type of repeaters that will only perform the delay correction if the input delay is less than the nominal delay δ. Interesting parallels can be drawn between this work and the work described in [13], where the SURFING pipelining technique is introduced together with some circuit design for a soft latch and an edge-to-pulse converter. Differently from the present work, the main requirement for the SURFING interconnect technique is to keep the travelling edges together; in the case described here the edges must be kept separated by a given amount. Also in [13] the edges may or may not occur, while in the phase-encoding method all edges always occur when a correct symbol is sent.

Several possible implementations are possible for a repeater; some types of circuits can be distinguished.

- Latch-based design, where the outputs are controlled by latches which are on the "data path."
- ME-based design where the MEs are on the data path.
- ME-based design where the MEs are not on the data path.

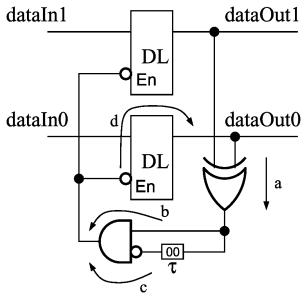

Fig. 3. Conceptual latch-based design.

The designs can be implemented both at transistor-level and at gate-level. Some examples for the design styles will be analyzed in the remainder of this section.

We introduce some measures to rank different designs: the *capture range* κ of the repeater is the set $\kappa = [\delta_{\min}, \delta_{\max}]$ of time separations at the input of the repeater that the device will be able to stretch back to the nominal value of δ. The *linearity* of the circuit corresponds to the linearity of the response of the design: given a range of time differences between two inputs, the linearity refers to the difference between the output delay between the signals and the input delay. The *latency* λ of the device is, intuitively, the time between the first input signal reaching the device and the corresponding output leaving the device. Finally, the *response time* ζ is the time between the first input reaching the device and the time limit before which the second event would not be delayed. For phase-locked loops (pll s) it is customary to provide the capture range (or "lock-in" range) as the difference between the maximum and the minimum frequency the pll will lock onto, as the set is centred around the natural frequency of the loop. Following the same custom, we similarly define $\kappa = \delta_{\max} - \zeta$ (as $\delta_{\min} = \zeta$).

Several implementations of the device are possible, each with different issues to be addressed. In particular, we can distinguish between "analog" implementations of the repeater and "digital" implementations of the device; in particular, this distinction is based on the generation of the pulse which stops the late arriving signal from propagating to the output. The analogue implementations rely on analogue differentiators to generate the pulse; these differentiators are built using series capacitors. In the case of digital implementations the differentiators are implemented using gates. In this article we will focus in particular on some digital implementations of the repeaters.

In the remainder of the section, all the simulation results are normalized to FO4 delay. The experimental setup included input and output inverter buffers; the latency shown in the graphs includes these buffers. "Rise" and "fall" in the results refer to the inputs shown in the figures, e.g., after the input buffer, to simplify analysis.

A. Latch-Based Design

A conceptual idea of the device is given in Fig. 3. The two input signals are fed into a pair of latches; the first input to propagate through one of the latches will cause a pulse at the "enable" input of the two latches, in turn causing the other signals to be prevented from propagating through the latch for a given

time. As can be seen from the simple example, two issues become of primary importance in the design of such devices: first, the latch of the signal arriving "late" should not enter metastability; second, which is correlated to the first, the time between an event triggering the pulse and the pulse reaching the "enable" input of the latches should be minimized in order to capture as many events as possible. In fact, if the latter condition does not hold, the late arriving signal will propagate through before the latch has been able to prevent this from happening.

The following description of the system is based on Fig. 3, but can be adapted to other designs. Define as d_{setup} the setup delay of the latches and as d_q the delay $D \rightarrow Q$ of the latch. From the figure, the letters a-d represent the path delays of the indicated paths; in particular, d represents the delay between the latch enable being released and the output being generated. Assume that two edges are travelling on the link and were generated with delay δ between each other and arrive at the input of the circuit with delay $\delta_{\text{in}} < \delta$.

The latency of the device is trivially $\lambda = d_q$. If

$$\delta_{\text{in}} < \lambda + a + b \tag{1}$$

the second edge will reach the output of the latch before the pulse is generated and therefore the circuit will have no effect. If

$$\delta_{\text{in}} \in [\lambda + a + b; \lambda + a + b + d_{\text{setup}}] \tag{2}$$

the latch may enter metastability, as the second edge will reach the latch during the time the pulse is being generated and will not respect the restriction imposed by the setup time of the latch. Finally, when

$$\delta_{\text{in}} > \lambda + a + b + d_{\text{setup}} \tag{3}$$

the latch will prevent the second edge to propagate until the pulse is completed.

The response time ζ will, therefore, be

$$\zeta = \lambda + a + b + d_{\text{setup}}. \tag{4}$$

The nominal value of δ at the output will be equal to

$$\delta = a + \tau + c + d. \tag{5}$$

The upper bound beyond which the device does not influence the path delays is going to be $\delta_{\max} = \delta + \lambda$; in fact if $\delta_{\text{in}} > \delta + \lambda$ by the time the second event occurs the latch has been released. Note that if the input time separation is between δ and $\delta + \lambda$, the output will still be δ. The capture range κ will, therefore, be

$$\kappa = \delta_{\max} - \zeta = \tau + c + d - b - d_{\text{setup}}. \tag{6}$$

These equations can be used to determine the value of τ given δ and a set of requirements. This first analysis shows that the value of δ must be chosen in such a way that, even when the phase is maximally corrupted, the condition expressed in (3) is always respected. This can turn out to be a significant limitation to the speed of the circuit.

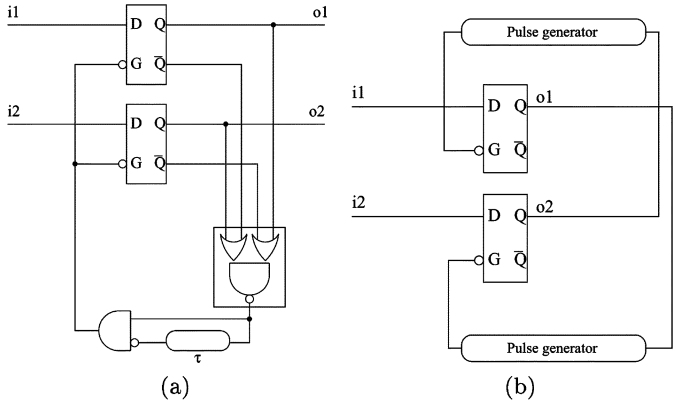

Fig. 4. Latch-based implementations. (a) Single pulse implementation. (b) Dual pulse implementation.

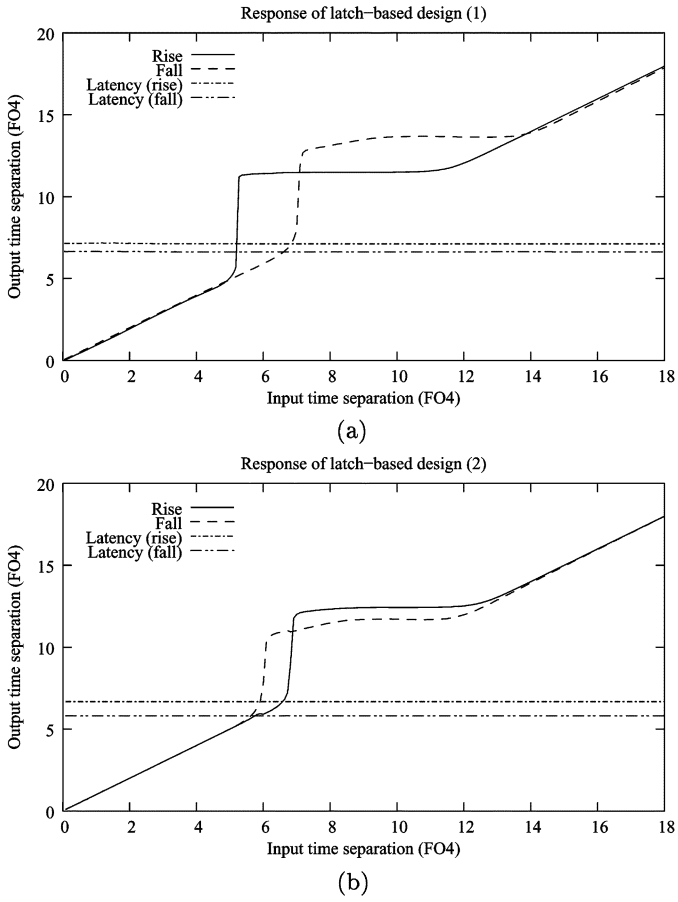

Fig. 5. Latch-based designs simulation results. (a) Single pulse generator. (b) Dual pulse generator.

Based on Fig. 3 two possible implementations can be obtained as shown in Fig. 4, according to the way the pulses which control the latch clock are generated. In Fig. 4(a) the pulse is generated as in Fig. 3, using an XOR gate, which has been reduced to an OR-NAND gate thanks to the presence of Q and $\overline{Q}$; in part (b), instead, each output generates an independent pulse to control the other latch. The two designs have very similar responses (Fig. 5); the expected output δ was around 12 FO4 delay, as this turned out to be the minimum time separation obtainable using a gate-level pulse generator and the available

latches. The device responds correctly: when the input time separation drops below 12 FO4 the device pulls the edges apart so that the output time separation is preserved at the output. The design fails when the input time separation drops below a minimum value (1). The latency of the device is around 6 FO4 delay for rising and falling edges. For the purposes of "taxonomy" this type of design is "early propagating," as an output is generated without waiting for the second edge to arrive.

The main advantages of this type of design is the relative simplicity of implementation. The gate count is relatively small and allows the repeater to be easily tuned to the required specifications. Also it would be relatively easy to scale up the design for multiple-rail implementations, simply by OR-ing all the XOR gates between pairs of wire and controlling a single pulse for all the latches on all wires. The latency of the design is also relatively low so that several repeaters can be placed along a line. Finally both design exhibit good linearity and the output curve are "flat" at the expected range. However, the capture range of the device is limited: both implementations stop working around 6 FO4 delay; around that point the risk of metastability in the latches increases until the devices fail. This type of design is therefore appropriate if the nominal δ of the link is in the order of tens of FO4.

IV. Fault Detection

An interesting aspect of this system is that fault detection is simplified thanks to the inherent nature of the transmission: for instance, stuck-at faults can be detected by observing the pattern of changes in the spacer. As the spacer alternates continuously between two possible spacers, the device can detect a stuck-at fault if the spacer alternation is not observed. Note that in that case, no data is received (as possibly one of the lines has not switched); in the case of two-wire communication, this could cause a problem as such a fault can be seen as a failed data transmission: the receivers could therefore throw an exception requesting the sender to repeat a transmission when none was previously performed. These issues can be resolved at design time (a time-out mechanism can be included, for instance); however, a common trait of any solution would be the inclusion of an accumulator which would "log" all the occurrences of a spacer exception and be reset at the occurrence of a correct sequence of spacers. If the sum of all exceptions exceeds a given value, the device would recognize the presence of a stuck-at fault.

In addition, the phase encoding mechanism allows transient faults to be filtered out rather than detected, thus reducing latency in case of faults. Consider for instance Fig. 6 where a link is implemented using dual-rail alternating spacer protocol and phase-encoding. In the top image a fault appears on one of the lines during data: in this case a *sp1* is detected after an item of data. Then an additional (wrong) item of data is detected and finally another *sp1* spacer is recorded. Only at this point the logic recognizes the presence of a fault and can request a retransmission of the data. A similar transient fault in the phase-encoding implementation is instead filtered out thanks to the inherent structure of the protocol: as only one wire presents a transition not matched by the other wire, the system ignores the fault. Note that in the previous case the fault is happening when the data item is present; although a fault during the same time

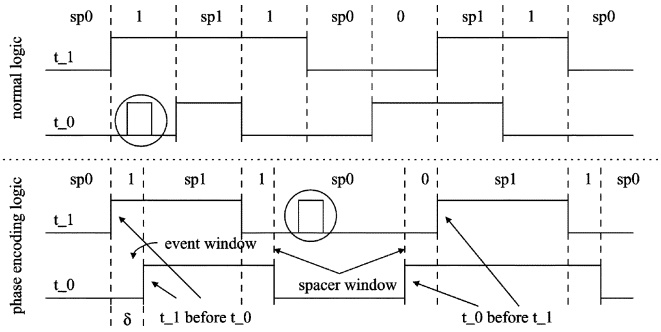

Fig. 6. Example of faults in level-based and phase encoding protocols.

in the phase-encoding protocol would also produce an error, the protocol aims at minimizing the window when an item of data is present on the lines.

We can identify in the context of phase-encoding *event windows* and *spacer windows*: the first denotes the time when there is an imbalance between the lines, the second indicating the time between two symbols when the wires are idle. We can then identify three types of transient faults, shown in Fig. 7, depending on the temporal position of the fault. Type-1 faults start and end inside the event windows; Type-2 faults appear during the spacer window. These faults are ignored by the protocol, although some Type-1 faults cause the receiver to require more time for the decoding as the input time separation is corrupted. We therefore distinguish between Type-1a faults—which do not cause significant increase of decoding time at the receiver—and Type-1b faults. Type-3 faults appear across event and spacer windows; in this category appear both faults that "mask" one of the transitions so that it could appear later or earlier than expected and faults that invert the sequence of transitions. The former are similar to the Type-1 faults described above and are ignored if the phase corruption does not increase the resolution time of the receiver beyond an acceptable limit; we therefore distinguish between Type-3a and Type-3b faults in the same way as for Type-1 faults. The latter we label as Type-3c faults and cannot be detected. More details about this limit is provided in Section V.

V. Bandwidth and Bitrate

The bitrate of the link is dictated by three factors: layout/process issues, number of wires and speed of the receiver to correctly decode the phase relationship between the input signals and produce an output. In [5] the relationship between bandwidth, δ and process parameters is described; in [7] the discussion is extended to the multiple-rail case. In both papers the analysis refers to the case where the MEs used at the receiver are built using a pair of NAND gates (SR latch) and a metastability filter [6]. We refer the reader to the mentioned works for a more accurate characterization of the bandwidth for phase encoding using those devices. In this work we generalize the analysis to the case where the input separation at the receiver is such that the device can avoid metastability. In this case the inequality which governs the frequency of the link is

$$T > n\rho T + t_{\mathrm{ME}}(t_{\mathrm{rise}}) + \lambda(n) \tag{7}$$

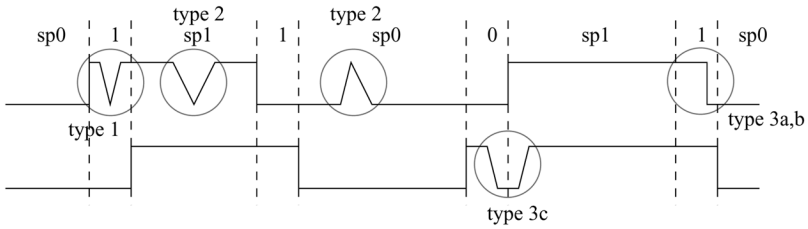

Fig. 7. Transient fault model in phase encoding logic.

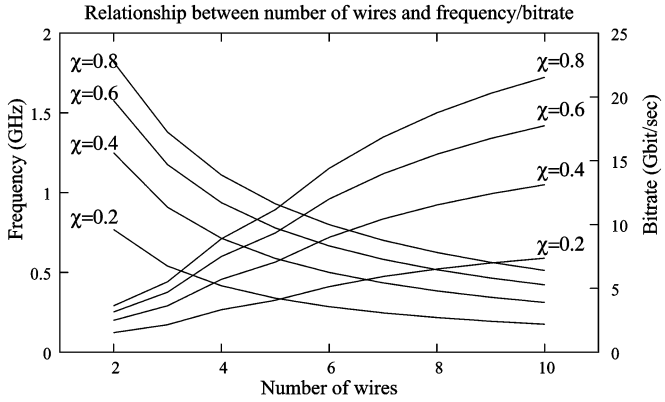

Fig. 8. Relationship between number of wires, frequency, bitrate, and link quality.

which is only valid if

$$\chi\rho T > t_{\mathrm{ME}}(t_{\mathrm{rise}}). \tag{8}$$

In these inequalities $\lambda(n)$ is the latency of the receiver $\rho = \delta/T$, T is the link maximum frequency, χ is the proportion of the original δ at the sender which reaches the receiver and t_{ME} is the ME resolution time, which is dependent on the rise time of the input signals t_{rise}. We can further simplify the inequality combining (7), (8)

$$T > t_{\mathrm{ME}}(t_{\mathrm{rise}})\left(1+\frac{n}{\chi}\right)+\lambda(n). \tag{9}$$

Fig. 8 shows the relationship between number of wires, frequency, bitrate and link quality in terms of χ, which we recall is the proportion of time separation at the sender between two edges which reaches the receiver; the results refer to 0.18-μm technology.

Note that the value of T considered until now only relates to the period of the frequency of the link where only one receiver is attached to the link. In the inequality $T > t_{\mathrm{ME}}(t_{\mathrm{rise}})(1 + (n/\chi)) + \lambda(n)$ the left hand size refers to the minimum time *between two symbols reaching a receiver*; in the case of Time Division Multiplexing (TDM) this time is equal to the frame length in time. The constraints for the maximum link frequency and those for the frame length can therefore be separated.

The maximum frequency f must be such that no overlap appear between two symbols; this is achieved imposing that the period $(1/f) = T_{\mathrm{link}}$ is greater than the symbol time. The latter depends on the number of wires and the value of $\delta = \rho T_{\mathrm{link}}$ plus an offset α for safe link operation; because of (8), $\rho T_{\mathrm{link}} > (t_{\mathrm{ME}}(t_{\mathrm{rise}})/\chi)$

$$T_{\mathrm{link}} > n\frac{t_{\mathrm{ME}}(t_{\mathrm{rise}})}{\chi}+\alpha. \tag{10}$$

The minimum frame length must accommodate the time needed for the receiver to decode the input. We can express the frame length as the time to send one symbol multiplied by the number of multiplexed receivers. If we assume that the receivers use rising and falling edges of the wave, the symbol time is $(1/2)T_{\mathrm{link}}$; we denote with σ the number of multiplexed receivers and thus obtain the expression for the frame length as $T_{\mathrm{frame}} = (1/2)\sigma T_{\mathrm{link}}$. We can now rewrite the above expression as $T_{\mathrm{frame}} > t_{\mathrm{ME}}(t_{\mathrm{rise}})(1 + (n/\chi)) + \lambda(n)$. Substituting

$$f < \frac{\frac{1}{2}\sigma}{t_{\mathrm{ME}}(t_{\mathrm{rise}})\left(1+\frac{n}{\chi}\right)+\lambda(n)}. \tag{11}$$

Inequalities 10 and 11 have to be satisfied simultaneously; the first indicates the condition for safe link operation from the point of view of accommodating the number of delays, while the second imposes that the link frequency cannot exceed a limit where the decoder cannot recover the data.

VI. Comparison With Other Protocols

A straight comparison between different communication models is always problematic: it is in fact difficult to make sure that the metrics used to estimate the performance of one or another technique are indeed fair. In this paper we limit to compare the proposed method with other communication protocols in terms of number of wires necessary to implement a channel, number of transitions to transmit a packet and number of symbols required to transmit a packet. The comparisons will be based around identical channel maximum frequency of operation.

In particular, it is important to note that the channel does not require a common clock between sender and receiver; also, there is no need to regenerate a signal clock at the receiver. Therefore, the obvious comparison would be with asynchronous communication methods, in particular m-of-n schemes.

Table I shows a summary of some encodings and a comparison with the phase encoding scheme illustrated here. Note that, even if the number of transitions per symbol appear to favor 1-of-n schemes, the overall number of transitions for a 128-bit packet example shows significantly different behavior. This, coupled with the availability of extra states which could be used to encode control signals, indicates an attractive feature

TABLE I
COMPARISON OF DIFFERENT ENCODING. PACKET LENGTH = 128 bits. RTZ = RETURN TO ZERO

Type of Link	Available States	Bits/Symbol	Extra States	Transitions/Packet	Symbols/Packet
4-rail Phase-enc.	24	4	8	128	32
1-of-4 RTZ	4	2	0	128	64
6-rail Phase-enc.	720	9	208	90	15
3-of-6 RTZ	20	4	4	192	32

of the scheme. Moreover, the number of symbols per packet is significantly smaller, reducing the channel usage. However the higher capacity of the phase-encoded link comes at the cost of a more complex gate-level implementation. For a "large" number of wires ($n > 6$) the sender and the receiver become difficult to design; this trade-off must be taken into account. Additionally, the use of more wires imposes a limit on the frequency of the link (as described in Section V) and therefore on its bitrate.

A comparison with synchronous links and clock-recovery schemes is complex due to the different metrics to consider. In general, if the frequency of the link is fixed and such to accommodate $n > 3$ wires for the phase-encoding protocol (see Section V), then the phase-encoding protocol has more capacity than an equivalent parallel synchonous link with n wires; as $\log_2 n! > n$ for $n > 3$. However, the fact that a delay δ is used between two edges could be exploited for synchronous operation, resulting in a high-frequency link with period of the order of δ; one example of such a system is presented in [14], where a serial link is described whose frequency is determined by delay elements, matched at the receiver to reconstruct the link clock.

Such systems however require highly matched delay elements; this matching can be obtained at the expense of increased static power consumption if, for example, current-starved inverters are used. The phase-encoding technique instead allows the individual wires to run at a lower frequency while still obtaining high bitrate. Also the matching required by the phase-encoding method is more loose, as the information is encoded only in the order of edges, rather than the absolute time delay between them.

VII. IMPLEMENTATION EXAMPLES AND RESULTS

In order to analyze the proposed method, implementation examples are described in this section. After the sender and receiver blocks, a complete system is described, comprising a 4-wire link.

A. Sender

The sender needs to encode the data presented as inputs into a sequence of transitions. In [5] examples are shown for dual-rail phase encoding transmitters, where delay elements are controlled by single- or dual-rail data lines and modulate the reference signal; Fig. 9(a) shows a block diagram of a sender in the case of n wires.

The sender encodes a symbol in a $n \times n$ matrix M whose elements correspond to internal control signals. These signals are used to control the phase encoding logic and are such that each row of the matrix corresponds to a wire and each column to a possible delay achievable on the wires. The n-wire phase

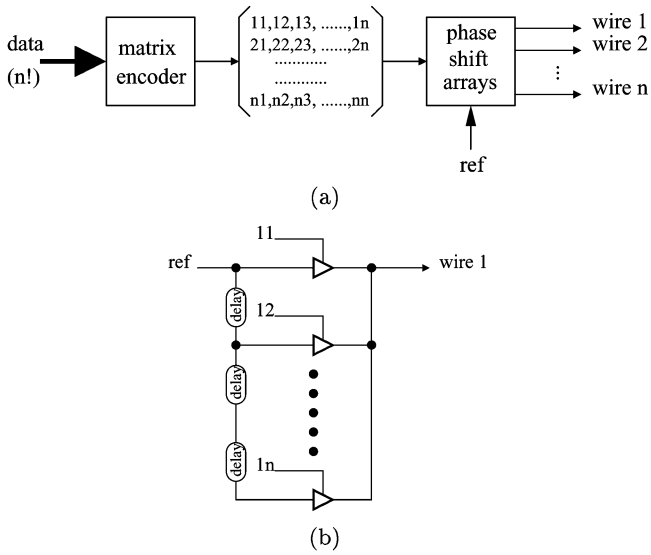

Fig. 9. (a) Block diagram of sender. (b) Delay chain.

encoding logic encodes $n!$ states into the symbol-dependent matrix according to the following algorithm. Each element of M is first set to zero. Each symbol is associated with a number in the sequence $S = \{0 \ldots n! - 1\}$; let the symbol to transmit be $s \in S$. The encoding logic decomposes the symbol in its *factorial base*, which, we recall, represents a number $k \in N$ as $k = d_1 \cdot 1! + d_2 \cdot 2! + d_3 \cdot 3! + \cdots + d_i \cdot i!$ where $i! < k$. The digits $d_{i..1}$ represent the index of the matrix row in the set of available rows at each iteration; the iteration represents the column. For instance, let $n = 4$ and $s = 17$ be a symbol to transmit. Its factorial base representation is $17 = 1 \cdot 1! + 2 \cdot 2! + 2 \cdot 3!$ and the order of digits to fill the matrix is $\{2, 2, 1\}$. Therefore, at iteration 0 element $m_{2,0}$ is set to 1. At iteration 1 the set of available rows is $\{0, 1, 3\}$: the digit 2 corresponds to row 3 and thus element $m_{3,1}$ is set to 1. Similarly $m_{1,2}$ is set. Finally an element of the remaining row, $m_{0,3}$, is set to 1. The final matrix is

$$M(17) = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

The algorithm can be implemented using banks of OR gates: at each iteration the fan-in of the gates changes and the output of the gates represents the matrix.

The phase encoding logic consists of a delay chain whose input is the reference signal and outputs are the n differentially delayed versions of the reference; it also comprises an array of tri-state buffers to direct the appropriate delayed version of the

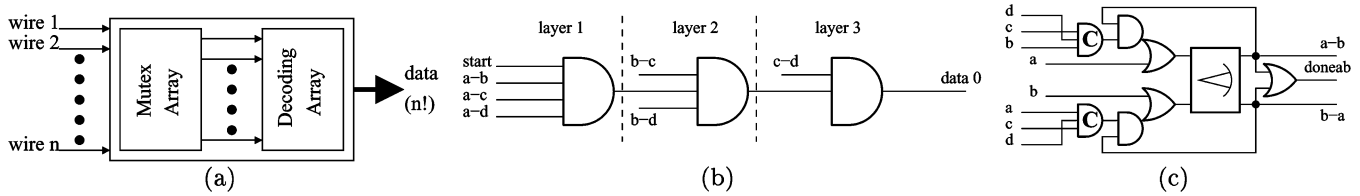

Fig. 10. Block diagram and details of receiver. (a) Block diagram. (b) Logic layers. (c) Enhanced ME.

reference to the corresponding wire. Fig. 9(b) shows this array for a single wire: the inputs to the tri-state buffers are controlled by the matrix M.

B. Receiver

The receiver for such a system is more complex than the sender, as it needs to identify and decode the sequence of transitions on n wires. MEs are employed in asynchronous systems to arbitrate between two requests avoiding metastability. Arbitration between multiple wires can be achieved using arbiter trees [15]; however this method is hardware-costly.

A different approach consists in employing arbiters for each pair-wise combination of wires; this array of arbiters will produce a binary output which is unique for each sequence and which is then fed to a decoder to extract the data. The block diagram of the receiver is shown in Fig. 10(a), in the ME array and the decoding array are shown. In this approach, for n wires an array of $\binom{n}{2}$ arbiters is required to cover all possible pair-wise combinations if only rising or falling transitions are used; in the case of both transitions being used the number of arbiters is double.

The logic used to decode the output of the ME array is organized into layers, each identifying the wire which wins a number of arbitrations. The sequence can be reconstructed analyzing the number of arbitrations each wire wins: the first wire will win all arbitrations with all other wires; the number of these arbitrations is thus $n-1$ for the first wire. The second wire will win all arbitrations with the other wires apart from the first and so on. Therefore, each layer identifies the wire which wins $n-l$ arbitrations: l is both the layer and the position of the wire in the sequence. This means that the number of logic layers is $n-1$, as the last wire will not win any arbitrations and can be identified by the last layer contextually to penultimate wire.

Compared with an arbiter-tree implementation this scheme reduces the latency of the receiver, as only one layer of arbiters is used. However, as the number of possible symbols increases (with n) the fanout of the layers also increases; this can result in a large area overhead. Alternative implementations of the receiver are the subject of future work.

C. Complete Example

Using the examples described above for the sender and the receiver, a circuit has been designed and simulated with $n=4$ wires, which we denote with letters *a-d*. We recall that the number of possible states is $n!=24$ and the number of arbiters is $2\binom{n}{2}$. Although both rising and falling transitions are employed to encode data in the implementation, in this section we only focus on the rising transitions for the sake of simplicity.

The sender was implemented as described above: the matrix controls the "enable" line of the tri-state buffers so that the reference signal is differentially modulated on the different lines. The matrix is obtained using one-hot encoding for the symbols; the "groups" described correspond to OR gates and the iterations correspond to logic layers. At each layer the fan-in of the OR-gate changes mirroring the "grouping" described above and goes from 2 to $n-1$; assuming that any arbitrary fan-in can be employed, the number of logic layers is $n-2$. For limited fan-in the total logic depth for n wires is $d=\sum_{i=2}^{n-1}\lceil \log_f(i)\rceil$ where f is the maximum allowable fan-in of the gates. For the case described $n=4$ and $f=3$, so the logic depth is $d=2$.

The receiver employs an array of MEs and the layered logic architecture described and shown in Fig. 10(b). Here the start signal is a control signal, generated when all MEs are settled down and all wires are at logic 1. *a-b* stands for wire *a* preceding wire *b* and so on for the other labels. The fan-in of each logic layer is the same as the number of arbitrations considered per layer plus an additional one which comes from the previous layer. In this example, there are three layers: the first layer is used to decide which wire wins three arbitrations; the second layer to decide which wire wins two arbitrations, and the third layer to decide which wire only wins one arbitration. The output of the logic is set to logic high if the associated symbol is detected [in Fig. 10(b)] this symbol is associated with the numeral 0, hence *data 0*).

The ME array is implemented including memory logic to achieve fault tolerance and guarantee correct operations. Fig. 10(c) shows the implementation of each element of the array, in particular the element responsible for arbitration between wires *a* and *b*. Apart from the ME in each element, there are two AND-OR gates, two three-input C-elements and one two-input OR gate. The two AND-OR gates are used together with the C-elements to construct memory elements to "keep" the outputs of the ME. When one wire wins arbitration and all other wires have switched, the corresponding AND-OR gate output keeps the output of the ME stable until the C-element changes state. This will only happen when all wires change state (at the next symbol). This mechanism guarantees that the output of the ME array is stable between symbols. Because memory elements are introduced, the completion detection of the arbitration is quite simple: only one OR gate for each element is enough because, after resolving, only one output of the ME will be logic high, whilst in idle mode the outputs will be both logic low.

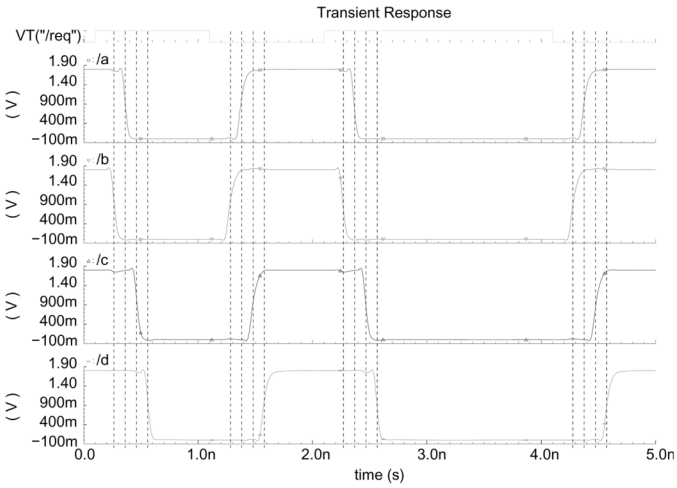

Fig. 11. Output waveforms at the sender.

This design respects the protocol described and accordingly responds to the faults described in Section IV. In the case of Type-1a and Type-3a faults the ME will not produce an error as the sequence of transitions has not been affected and the time separation is enough to ensure detection. In the case of Type-2 faults, thanks to the memory elements, the transitions appearing on one of the lines will be ignored, as the C-elements will not change state. Type-3c faults cannot be filtered out in the case of dual-rail phase-encoding schemes; multiple-rail implementation can be designed so that the probability of these faults to generate an error is minimized although this is not the case for the described implementation.

In the case of Type-1b, Type-3b faults the time difference between two edges is reduced in such a way that the resolution time of an ME is strongly affected. In the extreme case the transitions could be so close that the outcome of the arbitrations is random. In this case a "loop" can occur: consider wires *a,b,c,d* in this order and assume that wires *b,c,d* arrive at exactly the same time. The outcome of the arbitrations could be that *a* wins three arbitrations but all other wires win one arbitration: for instance, *b* may win with *c*, *c* with *d* but *d* may win with *b*. In this case the logic cannot recover the data, but is able to identify the fault and throw an exception, showing self-testing ability.

D. Simulation Results

The sender and receiver used in the four-wire link have been designed and simulated using a CMOS 0.18-μm technology and the Cadence toolkit. The simulation results show the sender and receiver working as expected.

Fig. 11 shows the waveforms of the sender. The encoded data correspond to the decimal 4; the corresponding transitions sequence *bacd*. Fig. 12(a)–(c) shows the waveforms obtained at the receiver when faults appear on the lines. The sequence of transitions in these figures is *abcd,* corresponds to the numeral 0. Type-1a and Type-2 faults are filtered out as can be seen from Fig. 12(a) and (b). Type-3c faults however will result in incorrect data being received: in Fig. 12(c) due to the Type-3c fault a numeral 6 is recovered.

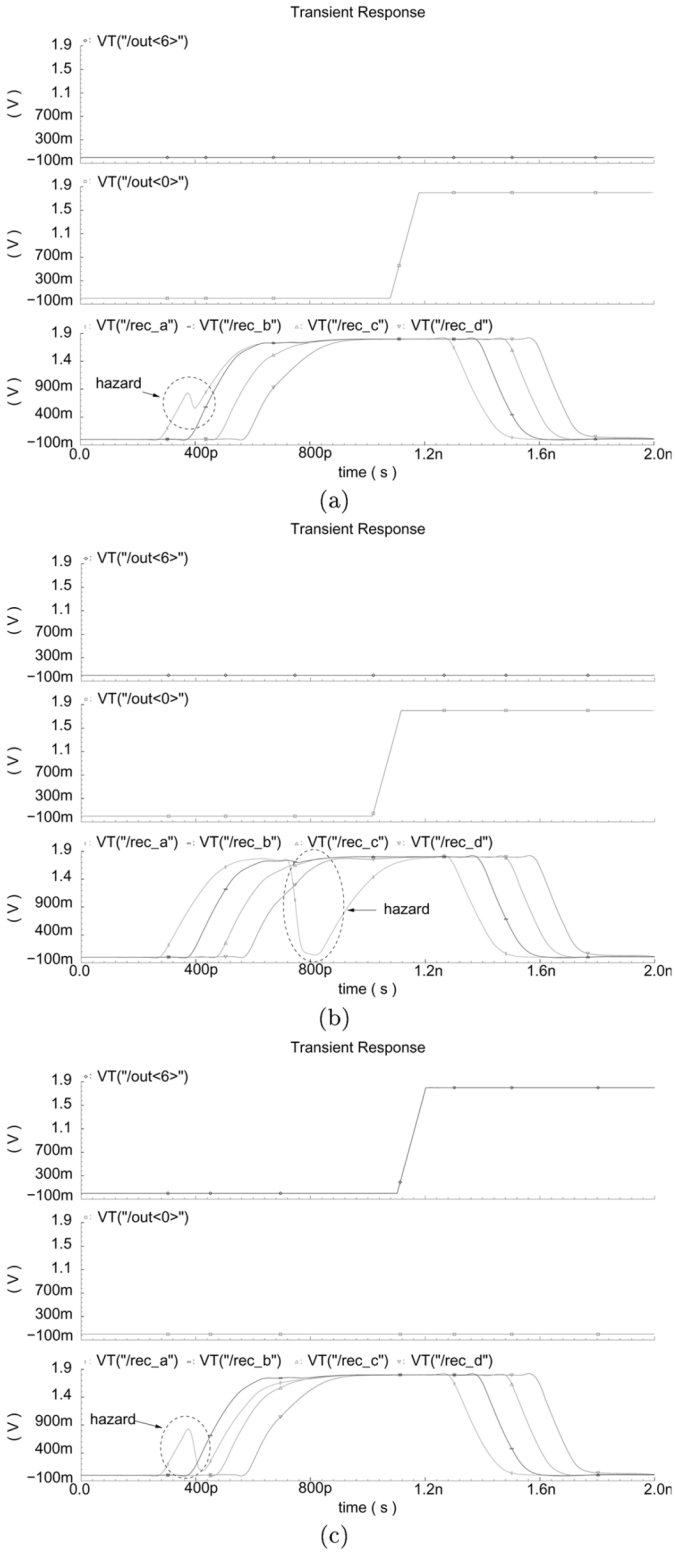

Fig. 12. Waveforms for faults. (a) Type-1. (b) Type-2. (c) Type-3.

VIII. Future Work

NoCs are the obvious application domain to investigate the full potential of this novel approach. For instance, the need to insert buffers along the transmission lines to regenerate the phase relationship goes hand in hand with the presence of buffers along the transmission medium of a NoC. The capabilities of

the system to support multiplexing is an additional useful property, together with the simplicity of error checking, which can be performed at various levels.

Some issues are under investigation and models and simulations are being developed, in particular in the areas of jitter estimation and selection of the optimal value of δ for reliable data recovery. The problem of jitter estimation is important: as the value of δ depends on the period of the reference signal (or in any case on the minimum time between two subsequent transitions on the reference line) and on the jitter, calculation of the jitter would lead to the identification of the minimum value for δ. However, jitter estimation is not a trivial task, as shown in the literature. Several options are under scrutiny at the moment, with promising results. Additionally, alternative design solutions for the receiving logic are being investigated.

Another subject of investigation is the optimization of wire length and buffer insertion. As described in Section III, it is a direction of the work to generate an algorithm, possibly to be automated, which would lead to the correct identification of optimal wire length and the inclusion of repeaters along the line. We also aim to investigate the addition of control logic on a bridge that would allow packet routing across the network employing this novel technique. This logic would have to read the packets, decode them and forward them appropriately.

Finally, work in under way to analyze various encoding schemes and a more accurate and thorough analysis and comparison of the scheme and other communication schemes. Different encodings can improve throughput of the link and reduce the error rate. At the same time, different encodings can increase the complexity of the receiver in terms of area consumption and decoding time. On the subject, a particularly attractive field of research linked with this idea is that of computational neuroscience; Thorpe and Gautrais [16] explore the use of rank order coding in the context of neural networks. Thorpe *et al.* [17] subsequently provide a review of strategies for spike-based activation of neurons. Neurons communication is modelled as sequences of spikes generated by neighbouring neurons; a neuron receiving these spikes can generate a spike if the sequence activates it. The sequence-recognizing algorithms used can be of interest and will therefore be investigated.

IX. Conclusion

A novel interconnection approach for SoCs has been presented together with some examples of implementation. The results show high robustness to transient faults of the type described (narrow-pulses) and relative simplicity of implementation. An important feature of the system described is the adaptability to a variety of environments (GALS, NoCs), achieved without the need for sophisticated circuitry. In fact, the system can almost be "plugged in" and work, as long as the synchronization protocol and the buffer stages are designed correctly.

The simulation results show that the circuit works as expected and has the ability to filter out interference. More accurate evaluation of jitter and identification of minimal event windows (possibly on-line) is under consideration. Together with the jitter introduced on the transmission lines, additional sources of jitter are in fact the delay elements themselves, particularly if a delay line is employed. A more analytical description of the design requirements is therefore being carried out, together with a more accurate probabilistic description of the effects of faults.

Future work aims to implement more complex protocols employing a larger number of transmission lines in order to increase throughput of the channel and increase reliability, automate the design process, identify the optimum wire length and repeaters number, and finally to devise reliable on-chip jitter estimation techniques.

Acknowledgment

The authors would like to thank L. Lavagno, D. Kinniment, and E. G. Chester for helpful discussions.

References

[1] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proc. Design Autom. Conf.*, Jun. 2001, pp. 684–689.

[2] A. Chakraborty and M. R. Greenstreet, "Efficient self-timed interfaces for crossing clock domains," in *Proc. 9th Int. Symp. Asynchronous Circuits Syst.*, May 2003.

[3] E. Dupont, M. Nicolaidis, and P. Rohr, "Embedded robustness IPs," in *Proc. 2002 Design, Autom. Test in Europe Conf. Exhib. (DATE'02)*, 2002.

[4] M. Nicolaidis, "Time redundancy based soft-error tolerance to rescue nanometer technologies," in *Proc. 17th IEEE VLSI Test Symp.*, Apr. 1999.

[5] C. D'Alessandro, D. Shang, A. Bystrov, and A. Yakovlev, "PSK signalling on SoC buses," in *Proc. Power Timing Modeling, Optim. Simulation*, 2005, vol. 3728, Lecture Notes in Computer Science.

[6] C. Molnar and I. Jones, "Simple circuits that work for complicated reasons," in *Proc. 6th Int. Symp. Asynchronous Circuits Syst.*, Apr. 2000, vol. 1, pp. 138–149.

[7] C. D'Alessandro, D. Shang, A. Bystrov, A. Yakovlev, and O. Maevsky, "Multiple-rail phase-encoding for NoC," in *Proc. 6th Int. Symp. Asynchronous Circuits Syst.*, Mar. 2006, pp. 107–116.

[8] D. Sokolov, J. Murphy, A. Bystrov, and A. Yakovlev, "Design and analysis of dual-rail circuits for security applications," *IEEE Trans. Comput.*, vol. 54, pp. 449–460, Apr. 2005.

[9] M. A. Kossel and M. L. Schmatz, "Jitter measurements of high-speed serial links," *IEEE Design Test Comput.*, vol. 21, pp. 536–543, Nov.–Dec. 2004.

[10] W. J. Bainbridge, "Asynchronous System-on-Chip Interconnect," Ph.D. dissertation, Dept. Elect. Eng., Univ. of Manchester, Manchester, U.K., 2000.

[11] J. Bainbridge and S. Furber, "Delay insensitive system-on-chip interconnect using 1-of-4 data encoding," in *Proc. 7th Int. Symp. Asynchronous Circuits Syst.*, 2001, pp. 118–126.

[12] C. D'Alessandro, A. Mokhov, A. Bystrov, and A. Yakovlev, "Delay/phase regeneration circuits," in *Proc. 13th Int. Symp. Asynchronous Circuits Syst.*, 2007, pp. 105–116.

[13] M. R. Greenstreet and J. Ren, "Surfing interconnect," in *Proc. 12th Int. Symp. Asynchronous Circuits Syst.*, Mar. 2006, pp. 98–106.

[14] S.-J. Lee, K. Kim, H. Kim, N. Cho, and H.-J. Yoo, "Adaptive network-on-chip with wave-front train serialization scheme," in *Dig. Tech. Papers Symp. VLSI Circuits*, Jun. 2005, pp. 104–107.

[15] C. L. Seitz, "Ideas about arbiters," *Lambda*, vol. 1, no. 1, pp. 10–14, 1980.

[16] S. J. Thorpe and J. Gautrais, "Rank order coding," in *Computational Neuroscience: Trends in Research*, J. Bower, Ed. New york: Plenum, 1998, pp. 113–118.

[17] S. Thorpe, A. Delorme, and R. Van Rullen, "Spike-based strategies for rapid processing," *Neural Netw.*, vol. 14, pp. 715–725, 2001.

Crescenzo Sabato D'Alessandro (S'06) received the B.Eng. degree in microelectronics and software engineering from Newcastle University, Newcastle upon Tyne, U.K., in 2001. He is currently working toward the Ph.D. degree in microelectronics design at the same university focussing on the area of on-chip interconnects for high-speed and signal integrity.

His current research interests include asynchronous VLSI circuit design, networks-on-chip, analog/mixed-signal design, and low-power design.

Mr. Alessandro has founded and chaired the Newcastle University IEEE Student Branch.

Delong Shang received the B.Sc. degree in computer science and technology from Nanjing University, Nanjing, China, the M.Eng. degree in computer engineering from the Chinese Academy of Sciences, Beijing, China, and the Ph.D. degree in computer systems and microelectronics from Newcastle University, Newcastle upon Tyne, U.K.

He is a Senior Research Associate with the school of Electrical, Electronic, and Computer Engineering, Newcastle University. His research interests include computer architectures, computer system and VLSI circuit design and testing, especially in asynchronous systems (VLSI circuits).

Alexander Bystrov received the graduate degree in electronic engineering from St. Petersburg State Electrical Engineering University, St. Petersburg, U.S.S.R., and the Ph.D. degree in testing of multilevel logic circuits from Napier University, Edinburgh, U.K., in 1986 and 1998, respectively.

He is a Lecturer in the School of Electrical, Electronic and Computer Engineering, Newcastle University, Newcastle upon Tyne, U.K. From 1986 to 1995 he worked as a Research Associate in the Department of Radio Systems of the St. Petersburg State Electrical Engineering University, studying on-line and off-line testing methods. Since 1998, he has worked in the Asynchronous Systems Laboratory , University of Newcastle. His research interests are modelling, visualization and design of asynchronous systems, arbitration, design of low-latency asynchronous circuits, on-line testing and security systems design.

Alexandre V. (Alex) Yakovlev (M'96–SM'07) was born in 1956 in Russia. He received the D.Sc. degree from Newcastle University, Newcastle upon Tyne, U.K., in 2006, and the M.Sc. and Ph.D. degrees from St. Petersburg Electrical Engineering Institute, St. Petersberg, U.S.S.R., in 1979 and 1982, respectively, where he worked in the area of asynchronous and concurrent systems.

In the period between 1982 and 1990, he held positions of Assistant and Associate Professor in the Computing Science Department. He first visited Newcastle as a Postdoctoral British Council Scholar in 1984–1985 for research in VLSI and design automation. After returning to U.K. in 1990, he worked for one year at University of Glamorgan. Since 1991, he has been at the Newcastle University, where he worked as a Lecturer, Reader, and Professor in the Computing Science Department until 2002, and is now heading the Microelectronic Systems Design Research Group (http://async.org.uk) at the School of Electrical, Electronic and Computer Engineering. His current interests and publications are in the field of modelling and design of asynchronous, concurrent, real-time and dependable systems on a chip. He has published four monographs and more than 200 papers in academic journals and conferences, has managed over 20 research contracts.

Dr. Yakovlev has chaired programme committees of several international conferences, including the IEEE International Symposium on Asynchronous Circuits and Systems, and is currently a chairman of the Steering Committee of the Conference on Application of Concurrency to System Design. He is a Member of the IET.

Oleg Maevsky received the M.S. degree in automated control and telemechanical systems from Frunze Polytechnic Institute, Frunze (now Bishkek), U.S.S.R., and the Ph.D. degree in computer engineering from Leningrad Institute of Electrical Engineering, Leningrad (St. Petersburg), U.S.S.R., in 1975 and 1985, respectively.

He worked as an Associate professor of the Software Department at the Kyrgyz-Russian Slavic University and International ATATURK ALATOO University, Bishkek, Kyrgyz Republic. In 2001, he was a Research Associate of Computing Science Department, VLSI group of the University of Newcastle, Newcastle upon Tyne, U.K. His research interests include VLSI design, analysis and synthesis of asynchronous (speed independent circuits, arbitration problems) and synchronous systems. From 2004 he has been an Engineer at Intel Corporation, Moscow, Russia.