An Asynchronous Interconnect Architecture for Device Security Enhancement

Simon Hollis, Simon W. Moore
Computer Laboratory, University of Cambridge, JJ Thomson Avenue, Cambridge, CB3 0FD, UK.
Email: {Simon.Hollis, Simon.Moore}@cl.cam.ac.uk

Abstract

We present a new style of long-distance, on-chip interconnect, based loosely on the asynchronous GasP architecture. It has a number of advantages over conventional designs, the most prominent being security enhancements, a reduction in the number of wires required, no need for clock distribution or packetization, and ease of composition.

We give some sample throughput and latency figures from simulation on a 0.18μm technology and show that it is viable for use with modern interconnect requirements, is of low complexity and has a lower area requirement than parallel interconnect over distances as short as 1mm.

1 Introduction

The International Technology Roadmap for Semiconductors indicates that on-chip interconnect latency and power consumption are increasing with respect to gate speed and power. Current CMOS technologies offer eight or more metal layers, but these incremental increases will never be able to keep up with the exponential increase in connectivity as predicted by Rent's Rule [4]. This problem is compounded by issues such as clock distribution, signal integrity and, of course, power concerns.

We present a modular interconnection system to manage these problems. Simulations show it has no synchronisation problems and offers a plug-in replacement for parallel interconnect buses at data rates close to 1Gbit/s. Communication links use dual-rail signalling, which balances power consumption. In particular, data dependent power consumption is minimised, improving security.

1.1 Global timing

Clocked circuit designs require that a clock signal arrives virtually simultaneously to all logic components. This low-skew clock distribution is well known not only to be very difficult to design [6], but also consumes lots of power.

Our scheme alleviates these difficulties by presenting two separate, synchronous interfaces: one to a transmitting circuit, and the other to a receiving one. In between, no global clocks are needed and we effectively use forms of local handshaking. This could be considered a Globally Asynchronous, Locally Synchronous (GALS) approach, and is a widely known solution to the problem of clock skew and multiple clock domains [11]. The major advantage of this approach to a systems designer is the ability to design all logic in a familiar, synchronous style and yet leverage asynchronous logic's ability to cross arbitrary clock domain boundaries reliably. This can aid composition, particularly in designs utilising multiple clock frequencies — a common method of reducing the power consumption of non-critical chip areas.

1.2 Security of devices

The dual-rail signalling scheme used reduces data dependent power, which can be measured in a number of ways in order to extract secret information. A full taxonomy of threats and abilities of attackers is given by Abraham et al. [1], and modern methods of attack are given in Anderson and Kuhn's excellent paper [3]. The most popular attack against devices today is not only powerful but also extremely cheap to implement. This is the attack of *electromagnetic analysis* (EMA).

We omit the full attack details, but the interested reader is invited to look at Agrawal et al.'s paper [2]. EMA has successfully read out data such as encryption keys from security access tokens and smartcards, as used in monetary payment systems [7]. Finally, note that EMA is classed as a *non-invasive* attack [3] — there is no tamper evidence, since no physical alterations are made to a device to perform EMA. This paper will present countermeasures to the EMA attack.

1.3 EMA and EMC

Electromagnetic Compatibility (EMC) is an additional concern for circuit designers. In many countries there is a restriction on the EM spectrum that a device is allowed to radiate. The techniques presented in this paper are intended to reduce a device's EM spectrum. Therefore, they should also help a device to pass electromagnetic compatibility testing.

1.4 Interconnect choice and circuit security

The key to a successful EMA attack relies on the fact that there is a correlation between the emitted EM spectrum of the device and the computation occurring at that point in time. Therefore, many authors have presented techniques for randomising execution [13], performing dummy data operations [9] and related ideas to reduce this correlation. We believe, however, that these approaches do not address the largest problem with regards to EM leakage on a modern chip: interconnect radiation.

Interconnect is, on a modern chip, by far the most power hungry component of any design, and this shows no sign of abating. Figures from Dally [5] show the transfer of 32 bits of data over 10mm will grow from consuming as much power as 25 ALU operations on a $0.13\mu\mathrm{m}$ process to more than 1,333 on $0.05\mu\mathrm{m}$. This rate of increase, compared to that of logic gate power consumption, puts interconnect at the heart of every modern design.

This power consumption disparity is especially significant for EMA; moving charge generates electromagnetic fields, and the larger the charge, the stronger the field. This is the reason for the focus of this work being interconnect EM emissions.

2 EMA countermeasures

Reduction of the utility of the emitted EM spectrum from interconnect can be achieved by any or all of the following countermeasures: reduction of the magnitude of the emitted EM radiation; reduction of the duration of the emitted EM radiation; reduction of the correlation of the emitted radiation and the data causing it, either in time or magnitude. We now present methods of effecting the first and third countermeasures above.

2.1 Magnitude reduction countermeasures

If a signal can be reduced to an insignificant magnitude then its frequency can stay well within the responsive frequency range of an attacker's probe with no ill effect. There are many methods of attempting this. Some are outlined below, with their drawbacks given in parentheses:

- Lowering the supply voltage (not always possible);
- Low voltage swing logic to reduce the current flow magnitude (signal integrity problems);
- Shielding with metalization (wastes valuable metal);
- Power balancing logic (expensive in area and power).

2.2 Frequency shifting countermeasures

When attempting to pick up an EM signal, probe-scope combinations are limited by their combined frequency response. A back-of-the-envelope calculation can be performed, recalling that impedance $Z = 2\pi fL + 1/2\pi fC + R$, where L, C and R relate to the probe-scope combination. We note that, since EM emissions occur at high frequencies, the L term dominates and, thus, impedance becomes directly proportional to the radiation frequency f. For a fixed sensor input, the higher this impedance becomes, the less voltage will appear across the scope input. Thus, by pushing the EM emissions higher up the frequency spectrum, we can reduce the signal available to an attacker. Practically, this involves running interconnect at higher frequencies.

2.3 Spread-spectrum countermeasures

We can perform both magnitude reduction and frequency shifting simultaneously by looking at spread-spectrum signalling. Spread-spectrum signals are ubiquitous in wireless communication as they provide numerous advantages. The one most interesting to us, though, is that they provide the ability for a signal carrying information to 'disappear' below an environmental noise floor. This is clearly useful to us as an attacker will be unable to eavesdrop on communication.

There are several methods of producing a spread-spectrum signal, with perhaps the most common being Code Division Multiple Access (CDMA) schemes.

3 Our interconnect approach

We now present our generic, long distance data interconnect system. We believe it has the ability not only to simplify long distance interconnect for many applications, but that it is especially suitable for use where an increased level of security is required. It is our belief that it offers a 'plug-in' replacement for existing point-to-point interconnection. As outlined below, it has lower complexity than any type of networked interconnect and a fixed power consumption per data bit. It is uni-directional, but two instances can be combined to produce a bi-directional link, and still maintain a small resource footprint.

3.1 Overview

Our interconnect offers many of the desirable properties illustrated in previous sections. We will see that dual-rail logic allows us to balance power, and therefore reduce data-related emissions. We utilise a high-speed and narrow core link, supplied by an asynchronous parallel-to-serial converter, which allows us to shift the frequency of emissions higher up the spectrum (see §2.2). It is based around previous work at Sun Microsystems on the *GasP* control method [14]. GasP was designed as a high-speed, asynchronous control path for controlling data passing though an asynchronous pipeline such as a micropipeline [15]. Its asynchronous structure gives it several advantages over a clocked rival such as the absence of a troublesome global clock signal, low idle-state power consumption, actual-case latency, and tolerance to power supply noise and voltage variation. The most significant concept of the GasP system is its use of a single wire to perform requests for *and* acknowledgments of data in the pipeline.

Our design uses this approach to reduce the wire count of our interconnect. Furthermore, using circuitry with local handshaking removes a large degree of correlation between the time transitions occur and the data they transfer, thus increasing security against EMA attacks.

3.2 The GasP control style

GasP signalling uses one wire to transmit both request and acknowledgment signals. To do this it utilises pulse-based bi-directional signalling on a so-called *state conductor* (the single wire). A downward transition on the state conductor is a forward-travelling request for the next pipeline stage, and an upward transition is a backward-travelling acknowledge signal for the previous pipeline stage.

This acknowledge signal can be though of as a reset phase whereby the state conductor is pre-charged to high via a p-type MOS transistor. The unusual part of the design is that the pre-charging is performed by the receiver rather than the transmitter, and signifies that the receiver is ready to receive new data. New data is sent by the transmitter pulling the state conductor low through an n-type transistor.

This arrangement of distributing the driving inverter not only allows events to be sent in both directions, but results in a reduced transistor count of two and the associated reduction in load capacitance, also aiding the speed of the circuit.

Necessarily for this protocol, both the transmitter and receiver drive their transistors with a pulse, just long enough to establish the voltage level on the state conductor. Longer pulses would result in an increased cycle time for no benefit.

Table 1. Dual-rail Semantics

Wire Values	Logical Meaning
`00`	Idle/Invalid Data
`01`	Logic 0
`10`	Logic 1
`11`	Undefined/Error

Table 2. Inverted Semantics

Wire Values	Logical Meaning
`00`	Undefined/Error
`01`	Logic 1
`10`	Logic 0
`11`	Idle/Invalid Data

3.3 Dual-rail logic

Dual-rail logic is a standard paradigm for asynchronous logic designers (see [12] for a comprehensive overview). Its main benefits are constant power consumption, regardless of the data value being transmitted, and the fact that data validity information is embedded in the data, so no additional control signals are needed in the forward direction. It uses two wires to transmit one bit and it has previously been recommended to help boost security against EMA [10], mainly due to its fixed power consumption, which can be considered as reducing the correlation in magnitude between data and EM emissions. Its semantics are given in Table 1.

The downside of dual-rail logic is that it requires two wires for every data bit being transferred — something that appears to contradict our earlier argument on the need to minimise complexity and wiring for power reasons. However, once the wire reduction of using a GasP-based approach is taken into account, our wire total is unchanged from a standard design. Yet we have reduced the power consumption and the level of compromising emissions, which is definitely a win.

A side-effect of using self-timed logic is that it is able to run freely of any main clock frequencies in a device. This is potentially very useful on slow devices, for example security devices, as we can run our interconnect much more quickly than the core. From the point of view of wanting EM emissions at as high a frequency as possible, this is clearly advantageous and comes for free with our scheme. Equally advantageous is the ability of self-timed logic to cross arbitrary clock domains. Our system can be used whenever this occurs, and may even aid power saving techniques, such as voltage islands or multiple clock domains, by transparently connecting these different areas.

3.4 Dual-rail GasP

The prototype implementation of our interconnect system deals with 8-bit wide data words, being input and output in parallel. It can be broken up into modular sections as illustrated in Fig. 1. We start on the left hand side with a parallel array of synchronous data latches and a synchronous buffer idle control signal. Between the input and output latches, the signals are all dual-rail (see §3.3), but those after and including the distributed inverter use return-to-one rather than the normal return-to-zero signalling (see Table 2). This helps them to fit in with our GasP style approach which utilises the high state for idle.

4 Implementation

Our design, as mentioned in the previous section, is based on the GasP system. In GasP, meaning is attached only to transitions and not actual logic values. Furthermore, all valid data consists of a downward transition.

As such, our system can be thought of as a pulse-based one, where each pulse could be considered a mini data packet, just long enough to maintain signal integrity of an edge through the system. Since the system waits for a full transmit buffer before firing data across the interconnect, it can be thought of as a 'burst mode' system. This entails further security advantages as bursts produce a wide, flat, spread EM spectrum.

As alluded to, our system presents locally synchronous interfaces to its input and output environments. These are illustrated at the edges of Fig. 1. An input environment observes a control signal, provides a fresh data word, and then asserts a `new_data` signal, latching in new data. Following observation of a `new_data` signal, the system proceeds to convert the data into a serial stream using a tree of 2-to-1 multiplexers (MUXs) in the order labeled in our figure as $d0 \dots d7$. We use this approach rather than a shift register because it decouples the timing requirements of the synchronous data provider and the asynchronous multiplexing.

Upon reaching the root multiplexer node, data embarks on its journey over long distance wiring using dual-rail GasP. Following its traversal of the long wires, data is ejected into a tree of demultiplexers (DEMUXs), at the leaves of which is an array of latches capable of taking the pulse form that the data is currently in and converting it to a more normal, level-based, one for consumption by the receiving module. Once all the latches are full (i.e. a full word has been received), the receiver is signalled synchronously with a `data_valid` signal and a synchronous `acknowledge` signal is waited upon before the latches are cleared and the next word is sent. To prevent metastability problems with the synchronous receiver, the `data_valid` signal should pass through a synchroniser. Mean Time Before Failure (MTBF) calculations indicate that, on our $0.18\mu\mathrm{m}$ process, a dual D-flop synchroniser gives a MTBF of $> 10^{135}$ years, at clock rates up to 1GHz [17].

4.1 The dual-rail distributed inverter

The distributed inverter concept is at the heart of GasP systems, where it is used as a *state conductor*. With GasP this transmits a control signal, but we use its principle twice: once on each wire of a dual-rail connection, to signal not only the validity of new data, but also its value.

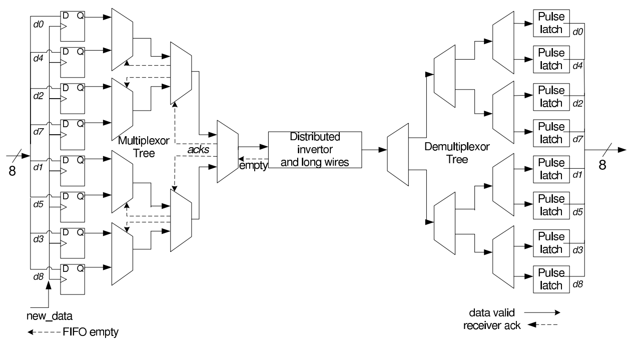


Figure 1. Our System Block Diagram

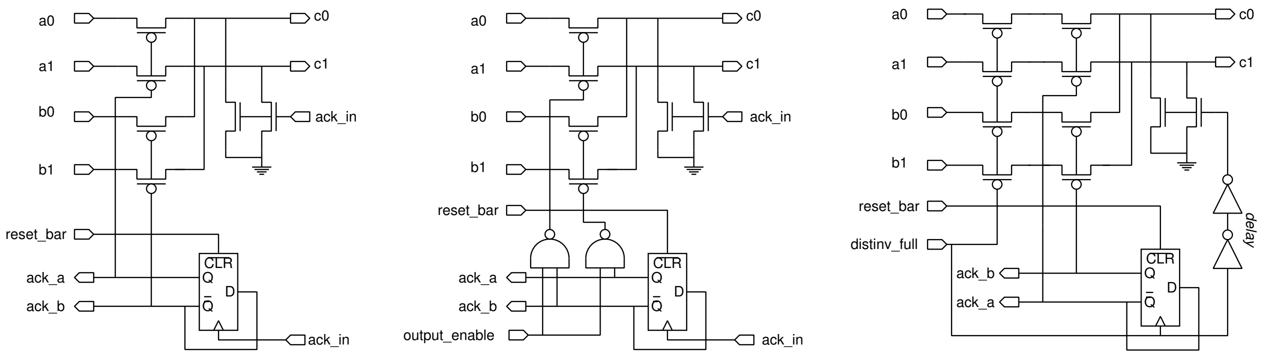
This scheme requires no more wires than those actively transmitting data and is an improvement over the normal dual-rail scheme. Detection of valid data on our distributed inverter output is trivial: we simply see if either data wire is at a low value. If it is, we have valid data, else we are in the idle/reset state.

4.2 MUXs

Much work was put into attempting to make the multiplexers as lightweight as possible. As such, each MUX has the minimum functionality required. This led to the production of three different types of MUX, dependent on their location in the tree. All MUXs use conventional dual-rail encoding.

4.2.1 The vanilla MUXs

The bulk of the MUXs are of a very simple design, illustrated in Fig. 2(a), consisting two dual-rail inputs, `a` and `b`, one output, `c`, and p-type pass transistors to pass high signals (the only ones with data valid semantics). The selection of a path is controlled by a D-flop, with `q` low selecting the path from input `a`, and $\overline{\mathtt{q}}$ low the path from input `b`. Clocking of the D-flop is performed for the MUXs by a local `ack` signal originating from the next MUX downstream. This `ack` indicates that the succeeding stages have successfully configured a new path for the next data bit to be transmitted. Paths begin being valid at the root node (the node nearest the distributed inverter), and then ripple monotonically outwards, before finally accepting new data as the path becomes valid in its entirety. Acknowledgments also cause MUXs to return their output bits to the idle state (here `00`), completing the return-to-zero phase in preparation for reception of the next valid input.



(a) The Vanilla Multiplexer Block (b) The Leaf Node Multiplexers (c) The Root Multiplexer

Figure 2. Our Three Flavours of Multiplexer

4.2.2 The leaf MUXs

Whilst remaining similar for the bulk of their design, the MUXs utilised for leaf nodes (i.e. those next to the input latches) require some additional logic. We combine path enable signals with a global enable signal, originating from the input latch control logic (see Fig. 2(b)), which signals when the input buffer has been drained of all data, and prevents the next item being transmitted until the buffer has been refilled. Leaf MUXs also block on a full output buffer.

4.2.3 The root MUX

The third variety of MUX is that used at the root of the multiplexer tree. It is responsible for injecting the current data bit into the distributed inverter driver, at the transmitter end of the long interconnect wire (see §4.1 for more details). It must also generate handshaking signals for the other MUX elements. Handshaking is generated based on the state of the distributed inverter and the root MUX's internal state. Thus a cascade of handshakes and new path setups is caused in the remaining MUXs, leading to a new path being established, targeting the next data bit from the buffer. Since a pulse is needed to drive the distributed inverter input, we incorporate a pulse generation circuit on the output n-types of the root MUX.

4.3 DEMUXs

Originally, our demultiplexing elements were designed without any state-holding elements on their data paths. The hope was to minimise the latency of the overall buffer structure and thus minimise the {new data → reset} cycle time. However, at each level of the DEMUX tree closer to the output, the necessary response speed need only be half that of the preceding one. As we may wish to make non-root DEMUX stages as slow as possible, for area and power reasons, we need some speed decoupling between DEMUX tree tiers. Furthermore, if we introduce local state-holding elements, the addition of decoupling will allow us to short-circuit the majority of the original control path, allowing the root DEMUX node (through which all data must pass) to have a very fast cycle time, pushing up the throughput. Therefore, we chose to include local state-holding elements in each DEMUX stage's data path. We use a standard D-type latch, the static nature of which allows each DEMUX to tolerate arbitrary inter-stage delays. We need only a single design for our DEMUXs, requiring no `acks`, since they have a *push* logic behaviour [12]; they are assumed to be fast enough to always accept data given to them, unless blocking when the output latches are full.

We now turn our attention to how the steering of the data path is performed. Since the only interesting events we have in our system are those which transition from high to low, we need only be concerned about transmitting logic `0`s. Therefore, we choose a design utilising n-type pass transistors to produce our path selection circuitry. We also include a global reset signal.

Clocking the D-flop is perhaps the most interesting aspect of the DEMUX, and is very simple given our protocol. We know that when we see a transition from a `11` input state to a state with either input `0` then we have gained new data. We can use this to disable the current data path, reset the input and enable an new path.

Input resetting is provided by strong p-type transistors on the input side of our n-type pass transistors. These are able to override any latched low signal coming from the previous DEMUX stage, signalling successful capture. This is then detected by the preceding stage's output, and causes the D-flops to cease driving their outputs via the latch $\overline{\texttt{preset}}$ signal. As with most of our design, the p-types are triggered by a pulse, generated by a pulse generator. The pulse's parameters are vital to correct operation, and it has been designed so that the initial signal passes through immediately, before being truncated or extended later. This configuration means

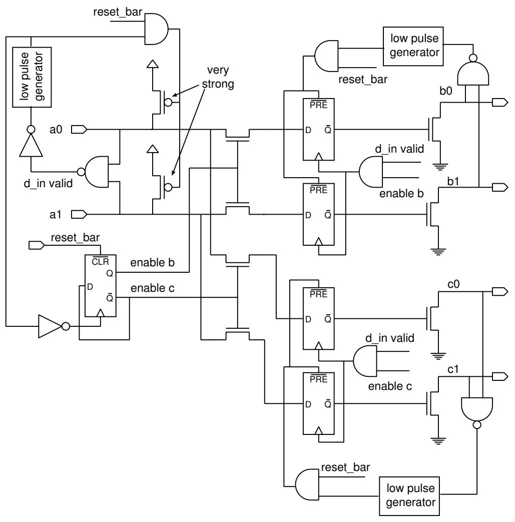


Figure 3. Our Demultiplexer Block

that it has virtually no impact at all on cycle time. Our DEMUX elements are shown in Fig. 3.

5 Methodology

Our results were gained from simulation of the complete system (as shown in Fig. 1) using the *hspice* suite [16] with a UMC $0.18\mu\mathrm{m}$ 1P6M process model. Spice netlists were generated automatically from the schematic tool *Electric*.

To compare directly with Ho et al.'s work (see §7), we took 3.8mm as the base length of our 'long' interconnect, which was modelled as a pair of parallel transmission lines using physical parameters from our process model. We chose to place these transmission lines on metal layer 6 as it was felt that such a rare, long distance interconnect would most likely be routed on this layer. Our simulations used a worst-case temperature estimate of $85\,^{\circ}\mathrm{C}$.

6 Results

The following results were gained from our simulations. We believe that, as a well characterised process, they are reliable and indicative of real silicon performance, although note that worst-case parameters were used.

6.1 Throughput

6.1.1 Distributed inverter

The core interconnection of our system is the distributed inverter, therefore system throughput is governed by its cycle time. Our simulations show that each bit takes an identical time to be transmitted, regardless of whether or not it is transmitted on the same wire as the previous one. This is highly desirable from a security perspective since we now have no time correlation with data values. The cycle time for a single data bit is 1.3ns (8 bits take 10.4ns), giving us a main interconnect data rate of 770Mbps, much higher than embedded systems' clock frequencies. Hence we hinder EMA by pushing the frequency up.

6.1.2 Environment

Due to additional control logic, the input buffer's external signals have a longer cycle time than the core interconnect. The good news is how small this overhead is: the time from the input environment signalling new data to receiving a 'input buffer empty' signal is 10.0ns. Note that this is *less* than the total for 8 cycles of the distributed inverter, because we can signal completion to the output environment simultaneously with transmitting the final acknowledgment (i.e. we need only really complete 7.5 cycles). Thus, at maximum throughput, our design's input and output frequency is limited by the core interconnect, and not its interfaces.

6.2 Latency

Throughput is not our only concern. A good interconnect system also has low latency. Our simulations show that the latency between a transmitter signalling that new data is valid to the receiver indicating that all data has successfully been received and is ready to be read out is 9.7ns. This may seem a little high, but ought to be acceptable for ASIC-based designs, and we believe it still represents good performance over a 3.8mm interconnect with no repeaters.

6.3 Area

We used the VST 0.18um VIP$^{\mathrm{TM}}$ Standard Cell Library to provide an estimate of the space required for our complete system. All flip-flops, logic gates etc. used in our design were matched to their equivalents in standard cell and, where this was not possible (e.g. for custom layout), an estimate of $1\mu\mathrm{m}^2$ per $1\mu\mathrm{m}$ of transistor gate width was used. We believe this to be a reasonable approximation that should give an upper bound on the area requirement. Size estimates for various configurations of our complete system are given in Table 3. We compare its area requirements for both logic and wiring to that of a standard parallel interconnect, all using minimum wire size and spacing. Full custom layout could give us a more compact design. Our system is competitive for wire lengths as small as 1mm, and an outright winner at the 3.8mm length we have been investigating, even if we omit the area of parallel interconnect's driving circuitry. This is due to the fact parallel interconnect requires eight wires for every two we use. Clearly, our system improves dramatically over parallel with increases in interconnect distance and wire width.

Table 3. Area usage (in $\mu\mathrm{m}^2$) for some designs

Configuration	Logic Area	Wire Area	Total Area
8-bit, simplex, our scheme, 1mm	5,984	1,320	7,304
8-bit, simplex, std. parallel, 1mm	> 0	6,925	$> 6,925$
32-bit, duplex, our scheme, 1mm	47,872	10,560	58,432
32-bit, duplex, std. parallel, 1mm	> 0	55,440	$> 55,440$
32-bit, duplex, our scheme, 3.8mm	47,872	40,128	88,000
32-bit, duplex, std. parallel, 3.8mm	> 0	210,520	$> 210,520$

Table 4. Our system performance v Ho et al.'s

Parameter	Our Design	Ho et al.
Data Width	8 bits	n bits
No. Data Wires	2	n
No. Control Wires	0	2
Throughput from 4 wires	1.6 Gbit/s	2.6 Gbit/s
Latency for a 3.8mm wire	1.30ns	0.7ns
Latency for a 5mm wire	1.33ns	1.3ns

7 Comparison to other interconnect designs

The most similar design to the one we present here is one presented by Ho et al. [8]. They use two GasP control lines to control a wide data path, enabling two requests and acknowledges to be in flight at once, halving latency. We believe that this is unnecessary and wasteful; the extra control line requires an extra wire. We believe we can obtain the same performance with the wires we already have.

We illustrate a comparison of their system and ours in Table 4. Our simulations were redone to use identical parameters of metal 5, a $0.18\mu\mathrm{m}$ process, and the same wire length of 3.8mm, except where noted. Our design degrades in performance slightly on the thinner metal layer. Our throughput may compare unfavourably to to their design but, for low bandwidth applications we have a clear win on wire count (our minimum is two, not four). For lines of 5mm or longer, we break even on latency. Further, our system appears to increase in latency with distance more slowly than theirs after this point.

8 Conclusion

We have presented a high speed serial interconnect to replace standard parallel on-chip interconnects used in ASICs. A multiplexer and demultiplexer tree structure to supply and sink data from the core interconnect has also been shown, and we have illustrated its lightweight nature. The high-speed serial core offers security enhancements against a EMA attack, and its asynchronous nature allows it to straddle multiple clock domains. Simulation in a $0.18\mu\mathrm{m}$ ASIC technology demonstrates that throughput approaches 1 Gbit/s, and we have seen that it presents a smaller area overhead to a parallel interconnect over distances as short as 1mm. It also compares well with other asynchronous approaches, such as Ho et al.'s [8], for narrow interconnections. Its modularity and serial nature makes it ideal for inclusion in ASIC designs, where ease of design and routing simplicity are key.

References

[1] D. G. Abraham, G. M. Dolan, G. P. Double, and J. V. Stevens. Transaction security system. *IBM Systems Journal*, pages 206–229, 1991.

[2] D. Agrawal, B. Archambeault, J. R. Rao, and P. Rohatgi. The EM side-channel(s). In *CHES 2002*.

[3] R. Anderson and M. Kuhn. Tamper resistance: A cautionary note. In *2nd USENIX Workshop on Electronic Commerce Proceedings*, Nov 1996.

[4] P. Christie and D. Stroobandt. The interpretation and application of Rent's rule. *IEEE Trans. Very Large Scale Integr. Syst.*, 8(6):639–648, 2000.

[5] W. Dally. Computer architecture is all about interconnect, Feb. 2002.

[6] E. Friedman. Clock distribution networks in synchronous digital integrated circuits. In *Proceedings of the IEEE, Vol 89, No. 5*, pages 665–692, May 2001.

[7] K. Gandolfi, C. Mourtel, and F. Olivier. Electromagnetic analysis: Concrete results. In *CHES 2001*.

[8] R. Ho, J. Gainsley, and R. Drost. Long wires and asynchronous control. In *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pages 240–249, Apr. 2004.

[9] P. Kocher. Using unpredictable information to minimize leakage from smartcards and other cryptosystems, Dec. 2001.

[10] S. W. Moore, R. Anderson, P. Cunningham, R. Mullins, and G. Taylor. Improving smart card security using self-timed circuits. In *Async 2002*.

[11] S. W. Moore, G. Taylor, R. Mullins, and P. Robinson. Point to point GALS interconnect. In *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pages 69–75, Apr. 2002.

[12] J. Sparsø and S. Furber, editors. *Principles of Asyncronous Circuit Design: A Systems Perspective*. 2001.

[13] E. Sprunk. Clock frequency modulation for secure microprocessors, Apr. 1995.

[14] I. Sutherland and S. Fairbanks. GasP: A minimal FIFO control. In *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pages 46–53, Mar. 2001.

[15] I. E. Sutherland. Micropipelines (the Turing award lecture). *Comm. A.C.M.*, 32(6):720–738.

[16] Synopsis(R). hspice.

[17] N. H. Weste and K. Eshraghian. *Principles of CMOS VLSI Design: A Systems Perspective (Second Edition).*