A GALS Router for Asynchronous Network-on-Chip

Pooria M.Yaghini, Ashkan Eghbal, and Nader Bagherzadeh
Center for Pervasive Communications and Computing
Department of Electrical Engineering and Computer Science
University of California, Irvine
{pooriam, aeghbal, nader}@uci.edu

ABSTRACT

A scalable asynchronous NoC router with lower power consumption and latency comparing to a synchronous design is introduced in this article. It employs GALS interfaces (synchronous to asynchronous/asynchronous to synchronous), imposing negligible area overhead to handle the Metastability issue. It is synthesized with the help of Persia tool, resulting in 23165 transistors. The power consumption and delay factor have been evaluated by means of H-Spice toolset in 90nm manufacturing technology. According to the experimental results the proposed asynchronous design consumes less power than synchronous scheme by removing clock signals. The imposed area overhead of asynchronous design is reported 36% higher than synchronous one.

1. INTRODUCTION

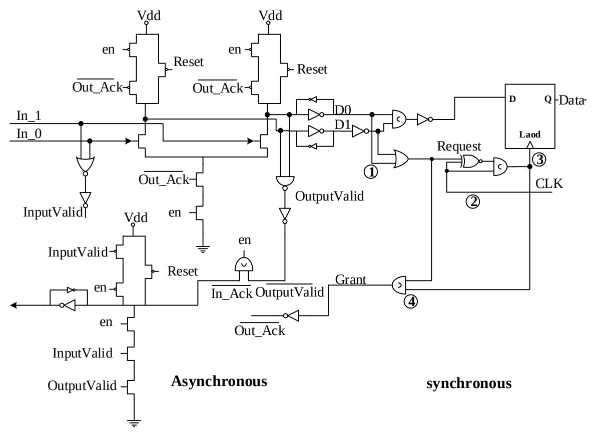
The capability of employing billions of transistors on a single chip results in the emergence of system-on-chip (SoC) and many-core architectures. However, SoC architecture is not scalable enough to support simultaneous packet transmission among the cores with a reasonable latency. The Network-on-Chip (NoC) has been introduced to accommodate better modularity, scalability, and higher bandwidth compared to bus-based connection [1, 2] to connect different Intellectual Property (IP) cores like GPUs, cache memories, and processors. [3, 4].

Different clock domains in large SoCs, is another challenging matter as the variety of connected IP-cores increases. A GALS (Globally Asynchronous Locally Synchronous Systems) technique is suggested to resolve it [5, 6] in which the IP-blocks employ the domain clock and they connect each other by means of request and acknowledge communication protocols [2]. So many research work has been experimented to implement a GALS design. Equalizing the frequency and phase attributes corresponding to the clock is one of them [7]. In reference [8] the distribution of a clock signal among various components with a constant phase difference has been examined. Employing similar frequency among various clock domains has been also evaluated [9]. Applying clock signals with different frequencies is experimented in some researches, while the frequencies are the factor of a constant integer [10, 11]. Most of the suggested remedies are time dependent, making them useless in global synchronization approaches. A totally asynchronous design would handle such issue and reduces the number of needed synchronous interfaces in a NoC network. An asynchronous NoC network is a GALS architecture if it employs asynchronous to synchronous interfaces and synchronous to asynchronous ones over communication points between each pair of IP-cores. Totally asynchronous NoCs have been designed in Nexus [12], ANoC [13], and MANGO [14]. The Nexus interconnect has a 16-port, 36-bit asynchronous crossbar that connects through asynchronous channels to clock-domain converters for each synchronous module [12]. Reference [13] has proposed a new asynchronous NoC architecture, providing low latency transfers. The proposed NoC protocol and its asynchronous implementation apply SystemC language. A clockless circuit of NoC router architecture has been presented in [14].

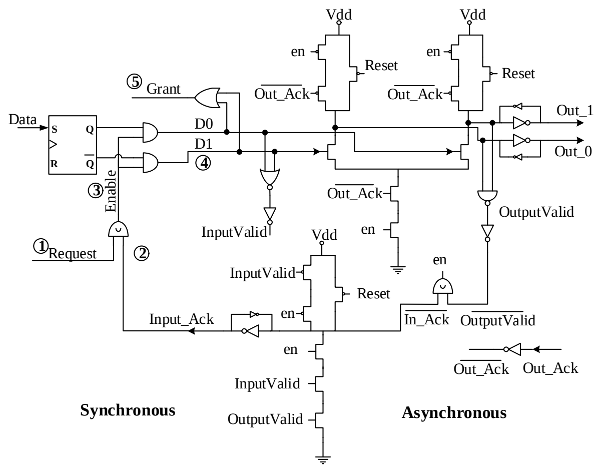
Delay and power effects of wires are considerable in many-core systems. Shorter period of clock cycles and the growth of die sizes make clock distribution implausible over the chip [15]. The clock skew is a major concern when a single clock is shared among various components. These facts minimize the total performance of a system or even results in storing an unexpected data inside registers. Symmetric and hierarchical clock trees have been suggested to mitigate the clock skew effects over the previous researches. However, such approaches increase the delay factor which is not acceptable in high speed applications [16]. A logical extension of the GALS approach into NoC design is an asynchronous (clockless) packet-switched network of clocked IP-blocks in which NIs (Network Interfaces) synchronization is needed. Such a method does not need a clock domain extension across the entire chip [17]. The novel contributions of this paper are:

- Proposing a novel scalable asynchronous NoC router to support GALS paradigm, with low power consumption and latency.
- Providing a light synchronous to asynchronous and an asynchronous to synchronous interfaces as a part of GALS design, offering lower possibility of Metastablity issue.
- Comparing physical performance parameters of two synchronous and asynchronous routers systematically.

MES '14, June 15 2014, Minneapolis, MN, USA
Copyright 2014 ACM 978-1-4503-2822-7/14/06...$15
http://dx.doi.org/10.1145/2613908.2613918.



(a) Asynchronous to Synchronous



(b) Synchronous to Asynchronous

Figure 1: GALS interface circuits

2. GALS INTERFACE

Scalability has become one of the most concerning matter as the complexity and the number of IP-cores soars. Global clock distribution nowadays is a formidable obstacle of scalability. To spread a global clock signal over a chip is not acceptable due to physical limitation, which can be resolved by applying the GALS method. Metastability is the major concern of employing GALS techniques, affecting data transmission throughput between synchronous and asynchronous regions. Any unpredicted delay variation of asynchronous designs might result in a serious malfunction in the synchronous side. The proposed architecture of this article is independent of delay factor as a Quasi Delay Insensitive (QDI) [18] model by applying Dual-rail coding method [6]. As there is no clock signal in this design, all of the data transmissions are put into action by the help of a four-phased handshake protocol. The asynchronous router is implemented with five bidirectional ports, applicable for Torus and Mesh topologies. The employed router architecture, implemented in CSP-Verilog, and its average latency of packet transmission is introduced in detail in [6].

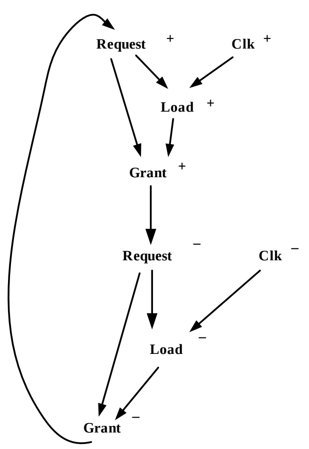
The only concern of the proposed design in this experiment is the NI component which is located in the middle of synchronous and asynchronous zones. On the other hand, in the GLAS design synchronization should be done at the boundary of the synchronous and asynchronous regions. Two special robust interfaces are proposed in this article to resolve such issue.

2.1 Asynchronous to Synchronous interface

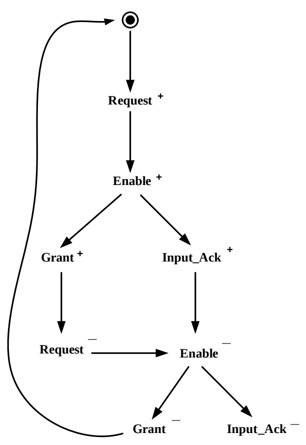
Figure 1a depicts the asynchronous to synchronous interface implementation with 4 sequence of data transmission. The STG related to such a design is illustrated in Figure 2a. The *Request* signal rises to announce its transmission demand toward a synchronous module once the asynchronous data of output lines *D0* and *D1* are ready. In synchronous side the *Request* signal is examined at each rising edge of the clock signal (*CLK*). The *Load* pin of a D flip-flap is activated if the synchronous module detects any request signal at the rising edge of clock. The asynchronous router is aware of reading its producing data when it finds both of the *Request* and *Grant* signal high. Such a situation makes the asynchronous router to feed the output data lines with neural data. The *Request* signal falls down once the neutral data appears on the data lines which results in inactivating the *Load* signal in the subsequent falling edge of the clock.

2.2 Synchronous to Asynchronous interface

The suggested synchronous to asynchronous design with its 5 sequence of data transmission is shown in Figure 1b. The corresponding STG with this design is depicted in Figure 2b. Synchronous to asynchronous interface waits for *Request* signal rising, deriving from synchronous module. The enable signal gets activated once *Request* signal rises while the value of *Input-Ack* signal is low. Activating *Enable* signal permits the transmission of synchronous data through the asynchronous module. The *Input-Ack* signal of asynchronous side and grant one in synchronous module rise as the incoming data from synchronous unit encodes into dual-rail mode. Activating *Grant* signal in synchronous module



(a) Async to Sync



(b) Sync to Async

Figure 2: STG diagram interface circuits

results in inactivating the *Request* and *Enable* signals and converting the asynchronous data lines into neutral values. The data path waits for the succeeding data by inactivating the *Grant* signal.

2.3 The Metastability Issue

The Metastability occurs because of failure in synchronization. The Metastability is the possibility of an unstable state to persist for a long (theoretically unlimited) period of time. In our design Metastability issue happens if a wrong data is read while the *Load* signal is enabled. In other words, newer flits are overwritten by proceeding ones because of wrong data sampling as a result of synchronization failure.

In this proposed interface, the probability of Metastability occurrence is very low (and is zero if we omit the probability of fault occurrence). As it is shown in Figure 1a data can be transferred if the *Load* signal of flip-flop is high. And as we use dual-rail coding, if the next data is supposed to be read, the data path should be neutralized (0, 0), which makes the existing C-Element in the data path to keep its previous data until the *Load* signal becomes low and alternately the sequence of four phase handshaking is finished. This will eliminate the probability of wrong sampling or overwriting while the *Load* signal is enabled.

3. EXPERIMENTAL RESULTS

The Asynchronous router is synthesized by Persia synthesis tool and then used Synopsys HSPICE tool to evaluate the power consumption and latency. Persia is a synthesis tool targeting template-based (PCFB-PCHB) QDI circuits. Input specification is in CSP-Verilog and output is a netlist of Persia standard cells. Persia uses standard layout tools for back-end as it encapsulates all the timing constraints inside manually laid out cells. Data Driven Decomposition is used for high level synthesis [19]. Persia uses PCFBs for its predefined templates. A PCFB template is an asynchronous buffer circuit that reads inputs, performs a particular calculation, and then writes the results to one or more of its output ports in each cycle of its operation.

The applicable throughput (inverse of the needed time by a flit to traverse a router) in asynchronous router is 1.1 GFlits/s. An 8-bit synchronous router [20] synthesized by Synopsys takes 17040 m^2 for the STMicroelectronics GPLVT standard cell library in 90nm process. For asynchronous router, the total silicon area of a 8-bit router is 23254 m^2, for the same fabrication process as illustrated in Table 1. The asynchronous router area is about 36% more than the synchronous router area. This is because of using dual-rail data encoding in asynchronous router, the area of the buffer, crossbar switch and specifically routing unit is about two times larger in asynchronous router than in synchronous router. Low power consumption is one of the advantages of employing an asynchronous design rather than a synchronous one [21]. In routing unit of asynchronous router, header flit is processed once, and then it will be inactive leading during transmission of data flits. This process sets the routing table and activates the arbiter unit to select the appropriate output channel. There is same story in the synchronous implementation. The routing unit is employed just as the receiving flit is recognized as a header, but it would be activated because of central clock. In other words, the routing unit transition is active while there is no header flit to route, leading to higher power consumption comparing to asynchronous design.

Table 1: Area evaluation

	Transistor	Area (μm^2)
Input Buffers	5975	6127
Crossbar Switch	6320	6390
Routing Unit	10870	10924
Router (Total)	23165	23254

Table 2: Energy (PJ) for A 32-flit packet

	Asynchronous		Synchronous	
Input Buffers	31	35%	128	47.6%
Crossbar Switch	25	27%	18.9	7.9%
Routing Unit	34	38%	120	44.5%
Router (Total)	91		266.310	

As it is shown in Table 2, routing unit in asynchronous design consumes lower power among the rest of components. The structure of switch component is simpler than input buffer and routing unit. This fact results in lower power consumption of this component in both Synchronous and Asynchronous routers. However, the switch employed in asynchronous architecture contains more gates due to its handshaking signals and its dual-rail coding architecture. Consequently, asynchronous router energy consumption is nearly independent of the packet content.

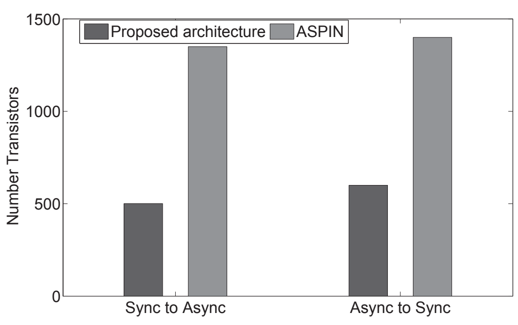
According to Table 3, both of suggested interfaces consume low power and has high throughput while imposing a low area overhead. Such properties, make applying the suggested interfaces plausible. Employing some extra gates is inevitable to the Metastablity issue in asynchronous to synchronous interface. Such imposed gates declines throughput and consumes more power comparing with the synchronous to asynchronous interface. These results also have been compared with a similar work which is called ASPIN [22]. The GALS NoC router is implemented with the instinctive behavior of asynchronous network in this experiment to compare with ASPIN.

The comparison between the proposed architecture and a similar asynchronous one is shown in Figure 3 in terms of area overhead and throughput. The area overhead of the proposed architecture is less than half of area consumption of ASPIN. Throughput of Synchronous to Asynchronous interface, depicted in Figure 3b, in the proposed design is higher than the similar one in the ASPIN design since it has extra transistors to prevent Metastability. However, reported throughput of Asynchronous to Synchronous interface in the proposed design is less than similar component of ASPIN architecture.

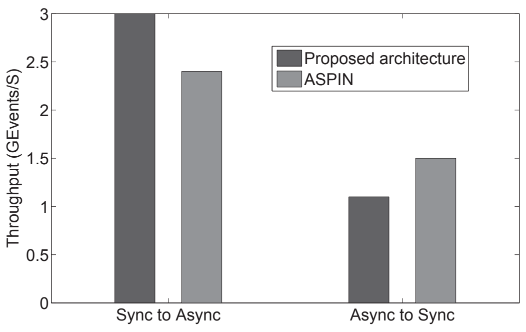
4. CONCLUSION

Table 3: GALS Interfaces Properties

	Power (μW)	Through. (MFlits/s)	Trans.	Area (μm^2)
Sync to Async	0.24815	3030	512	602
Async to Sync	3.39	989.4132	600	677



(a) Footprint



(b) Throughput

Figure 3: Proposed Architecture vs. ASPIN

A power efficient, scalable, and low latency asynchronous NoC router architecture has been introduced in this paper. It is composed of a buffer and switch components to store and transmit the flits (data flow)and a routing unit to direct each of incoming flits to the appropriate output channel (control flow). The routing unit in asynchronous design is more complex than the synchronous scheme due to handshaking signals but it consumes less power because of eliminating the permanent clock signals. The implemented buffer component in such a design reduces the delay of long wires, resulting in better throughput than other designs. Two GALS interfaces have been proposed as a solution of Metastability issue of applying GALS implementation. The experimental results show that the hardware overhead of suggested interfaces are negligible, which is 2.5% of the whole asynchronous NoC router. The asynchronous design needs 36% more area space than similar synchronous one. It also provides a 1.1 Gflit/s transmission throughput. The advantage of using the proposed GALS wrappers in NoC routers is to increase the throughput of NoC while consuming lower power.

5. REFERENCES

[1] Mohammad Hosseinabady, Abbas Banaiyan, Mahdi Nazm Bojnordi, and Zainalabedin Navabi. A concurrent testing method for noc switches. In *Proceedings of the Conference on Design, Automation and Test in Europe: Proceedings*, pages 1171–1176. European Design and Automation Association, 2006.

[2] P.M. Yaghini, A. Eghbal, H. Pedram, and H.-R. Zarandi. Investigation of transient fault effects in an asynchronous noc router. In *Parallel, Distributed and Network-Based Processing (PDP), 18th Euromicro International Conference on*, pages 540–545, 2010.

[3] M. Zolghadr, K. Mirhosseini, S. Gorgin, and A. Nayebi. Gpu-based noc simulator. In *Formal Methods and Models for Codesign (MEMOCODE), 9th IEEE/ACM Conference on*, pages 83–88, July 2011.

[4] Abbas BanaiyanMofrad, Gustavo Girão, and Nikil Dutt. A novel noc-based design for fault-tolerance of last-level caches in cmps. In *International Conference on Hardware/Software Codesign and System Synthesis*, pages 63–72. ACM, 2012.

[5] J. Muttersbach, T. Villiger, and Wolfgang Fichtner. Practical design of globally-asynchronous locally-synchronous systems. In *Advanced Research in Asynchronous Circuits and Systems. Sixth International Symposium on*, pages 52–59, 2000.

[6] Pooria M Yaghini, Ashkan Eghbal, Hossein Pedram, and Hamid Reza Zarandi. Investigation of transient fault effects in synchronous and asynchronous network on chip router. *Journal of Systems Architecture*, 57(1):61–68, 2011.

[7] A. Chakraborty and M.R. Greenstreet. Efficient self-timed interfaces for crossing clock domains. In *Asynchronous Circuits and Systems. Ninth International Symposium on*, pages 78–88, 2003.

[8] E. Nilsson and J. Oberg. Reducing power and latency in 2-d mesh nocs using globally pseudochronous locally synchronous clocking. In *Hardware/Software Codesign and System Synthesis. International Conference on*, pages 176–181, 2004.

[9] I. Miro Panades and A. Greiner. Bi-synchronous fifo for synchronous circuit communication well suited for network-on-chip in gals architectures. In *Networks-on-Chip, 2007. First International Symposium on*, pages 83–94, 2007.

[10] J. Mekie, S. Chakraborty, G. Venkataramani, P. S. Thiagarajan, and D. K. Sharma. Interface design for rationally clocked gals systems. In *Asynchronous Circuits and Systems. 12th IEEE International Symposium on*, pages 12 pp.–171, 2006.

[11] Uri Frank, Tsachy Kapshitz, and Ran Ginosar. A predictive synchronizer for periodic clock domains. *Formal Methods in System Design*, 28(2):171–186, 2006.

[12] A. Lines. Asynchronous interconnect for synchronous soc design. *Micro, IEEE*, 24(1):32–41, Jan 2004.

[13] E. Beigne, F. Clermidy, P. Vivet, A. Clouard, and M. Renaudin. An asynchronous noc architecture providing low latency service and its multi-level design framework. In *Asynchronous Circuits and Systems. 11th IEEE International Symposium on*, pages 54–63, March 2005.

[14] T. Bjerregaard and J. Sparso. A router architecture for connection-oriented service guarantees in the mango clockless network-on-chip. In *Design, Automation and Test in Europe.*, pages 1226–1231 Vol. 2, March 2005.

[15] Johnny ÃŰberg. Clocking strategies for networks-on-chip. In Axel Jantsch and Hannu Tenhunen, editors, *Networks on Chip*, pages 153–172. Springer US, 2003.

[16] E.G. Friedman. Clock distribution networks in synchronous digital integrated circuits. *Proceedings of the IEEE*, 89(5):665–692, 2001.

[17] Jens SparsÃÿ. *Future Networks-on-Chip; will they be Synchronous or Asynchronous? (Invited talk)*. 2004.

[18] California Institute of Technology. Computer Science Dept. (CITCS) and A.J. Martin. *Synthesis of Asynchronous VLSI Circuits*. California Institute of Technology, Computer Science Department, 1993.

[19] Mehrdad Najibi. Persia: A qdi asynchronous synthesis tool, 2008.

[20] A. Eghbal, P.M. Yaghini, H. Pedram, and H.-R. Zarandi. Fault injection-based evaluation of a synchronous noc router. In *On-Line Testing Symposium. IOLTS 2009. 15th IEEE International*, pages 212–214, June 2009.

[21] P.M. Yaghini, A. Eghbal, S.A. Asghari, and H. Pedram. Power comparison of an asynchronous and synchronous network on chip router. In *Computer Conference. CSICC 2009. 14th International CSI*, pages 242–246, Oct 2009.

[22] D. Lattard, E. Beigne, F. Clermidy, Y. Durand, R. Lemaire, P. Vivet, and F. Berens. A reconfigurable baseband platform based on an asynchronous network-on-chip. *Solid-State Circuits, IEEE Journal of*, 43:223–235, Jan 2008.