IPSJ Transactions on System LSI Design Methodology Vol. 1 58–66 (Aug. 2008)

Regular Paper

A Study of Multi-core Processor Design with Asynchronous Interconnect Using Synchronous Design Tools*1

Katsunori Tanaka,†1 Yuichi Nakamura†1
and Atsushi Atarashi†1

This paper presents a study of GALS (Globally-Asynchronous Locally-Synchronous) architecture multi-core processor design with asynchronous interconnects. While GALS is expected to reduce more power dissipation, it has not been the mainstream of LSI design yet, since there have been no mature design tools for asynchronous circuit design. For GALS design, we constructed a design flow based on general synchronous design tools, by specification of design constraints and configurations. Applying the design flow to an experimental multi-core processor GALS design including an asynchronous interconnect based on QDI (Quasi Delay Insensitive) model, we successfully obtained a netlist and layout, and proved that the flow works correctly, by netlist simulation with delay information back-annotated from the layout. Experimental results show the area, power and throughput of the asynchronous interconnect to indicate the impact by introducing GALS architecture instead of globally synchronous design.

1. Introduction

Power reduction is recently a serious issue on LSI design. There are a lot of approaches for low power design at various design levels.

Implementation of more than one clock domain is getting popular for the optimal tradeoff between power and performance. By minimizing the clock frequency in each domain for the required performance, then the power consumption of each domain, thereby the total power consumption of the chip, is reduced for the minimum. Today's multiple clock implementations, however, do not give the complete freedom of providing arbitrary frequencies to the domains, due to restrictions imposed by design rules and tools available today. For example, all the clock frequencies must be synchronized, or to be more exact, every clock must be configured to be an integral multiple of the base clock frequency. As a result, power about synchronization is wasted, since the domains are driven at clock frequencies higher than necessary.

In order to improve power consumption, GALS (Globally Asynchronous, Locally Synchronous) design method has been proposed to provide the minimum frequencies to the domains [1)–5)].

In GALS architecture, handshaking signals control the status of latches and flip-flops to guarantee consistent data transfer among a variety of clock domains.

While asynchronous logic circuits are applied to GALS architecture, no practical design flow has been established so far. The first concern about GALS design flow is design methodology of asynchronous interconnects. There are a few asynchronous VLSI design tool products from start-up companies [6)]. They are, however, still at early stage and require additional costs for purchase. A few papers are focused on the design flow for asynchronous circuits based on synchronous design tools. Kondratyev, et al., however, did not cover the physical synthesis steps like placement and routing [7)]. Ozdag et al. covered them, but did not discuss specific issues for clockless circuit design, such as delay adjustment [8)]. For GALS design flow, we can thus barely find some point tools, but not total design flow.

In this paper, we present a study of multi-core processor in GALS architecture, and a design flow for the GALS architecture based on design tools for synchronous circuits. Given RTL description as its input, our proposed design flow outputs layout information. It is applicable to a GALS design containing asynchronous interconnects.

In the design flow, we assume QDI (Quasi Delay Insensitive) model with multiple-rail delay insensitive encoding, since it is impossible to apply DI model to the design of most classes of asynchronous circuits [10)–12)]. The QDI model assumes no restriction in regard to the delays of the elements and wires, and we can be free from any design constraint resulting from the length of wires.

When synchronous tools are simply applied to the GALS design, they may produce incorrect design results, since they assume that the design is globally synchronous (GS). We have thus derived several design constraints to complete

†1 NEC Corporation
⋆1 This paper is based on Ref. 18).

the proposed design flow. These constraints can be incorporated well into the design tools for synchronous circuits.

We show an experimental GALS design, which consists of four synchronous processor cores and one synchronous data memory interface core connected by an asynchronous interconnect. We have obtained a netlist with layout information for the GALS design and have proved the correctness of the result with simulation. We also indicate several statistics such as core size, throughput and power dissipation to discuss the impact by introducing GALS architecture instead of GS design scheme. The power per area by the asynchronous interconnect is 64%, compared with its synchronous counterpart, which implies the potential of GALS design with asynchronous interconnects.

This paper is organized as follows: Section 2 explains QDI model asynchronous logic. Section 3 shows the design flow for synthesis and layout of GALS design composed of synchronous cores and asynchronous interconnect. Section 4 describes the result for an example design consisting of synchronous processor cores and a synchronous data memory interface core connected by an asynchronous interconnect, along with the evaluation regarding its area, power dissipation and throughput. Section 5 concludes this paper.

2. QDI Model Asynchronous Logic

In this section, we introduce QDI (Quasi Delay Insensitive) model for asynchronous circuit design. The term "asynchronous" is used to mention relationship among clocks. Instead, we use another term "clockless" for the circuits working without any clocks. Similarly, we use "clocked" for the circuits working with clocks. Corresponding to the term, "GALS", globally synchronous or clocked circuits are abbreviated as "GS" henceforth.

Delay insensitive (DI) model is an assumption on delays of cells and wires, and it assumes that the delays are positive values but unknown [12]. The assumption is completely different from that of clocked circuit design where the maximum and minimum of the delays are estimated prior to the design. If clockless logic circuits are designed on DI model, they work correctly, no matter how long the delays are. On the DI model, however, practical functionality cannot be realized, since only inverters, buffers and Muller's elements can be applied to the DI model based design [10].

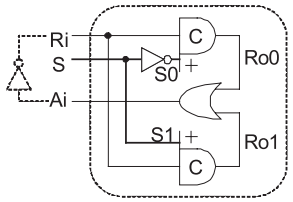

Fig. 1 Example: Non-DI clockless circuit structure.

Figure 1 illustrates an example of non-DI clockless circuit structure to imply difficulty of DI-based design. In this example, cells denoted as AND-like cells labeled with "C" are called asymmetric C-elements. An asymmetric C-element outputs 1 if its both input signals are 1. It outputs 0 if the input signal to the pin without "+" is 0. Otherwise, the C-element holds its state.

By using these C-elements with signal S, the example circuit makes selective handshaking. This circuit structure is found in clockless interconnects. If $S = 0$ when Ri becomes 1, the handshaking signal transitions are $Ri = 1 \rightarrow Ro0 = 1 \rightarrow Ai = 1 \rightarrow Ri = 0 \rightarrow Ro0 = 0 \rightarrow Ai = 0$. If $S = 1$, then $Ro1$ is involved instead of $Ro0$ in the handshaking. No matter how long the delays of the C-elements and OR gate are, this example circuit works correctly.

The example circuit is likely to malfunction, however, if the inverter on the fork from S has a delay. In the design of the circuit, either $S0$ or $S1$ is assumed to be 1 at any time, but $S0 = S1 = 1$ holds for the inverter's delay in $S = 0 \rightarrow 1$. If $Ri = 0 \rightarrow 1$ reaches the C-elements during the period of $S0 = S1 = 1$, then both $Ro0$ and $Ro1$ unexpectedly become 1. This observation implies at least that based on the DI model, it is very difficult to design clockless circuits realizing their functionalities by forks.

To achieve clockless circuit design with practical functionality, the QDI model thus extends the DI model with an assumption that delays on a fork are equal. Such a fork is called an **isochronic fork**. Thus, the assumption is imported as design constraints to the clockless circuit design on the QDI model.

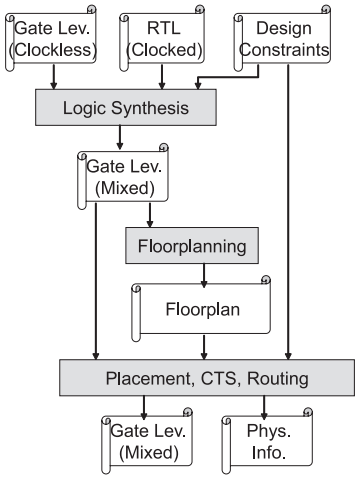

Fig. 2 Design flow.

3. Design Flow for GALS Architecture

In this section, we propose a design flow for GALS architecture with a clockless interconnect based on QDI model among clocked cores. The design flow we have established is compliant to the design flow for clocked circuits, as summarized in **Fig. 2**. In applying design tools for clocked circuits to the clockless modules in the GALS design, we have to deal with three difficulties, whose details will be discussed in Section 3.1.

3.1 Difficulties for Applying Clocked Design Tools to Clockless Modules

In this section, we introduce the three difficulties in applying design tools for clocked circuits straightforwardly to clockless circuits.

First, general logic synthesis tools destroy the functionality of the clockless modules in the GALS design by transforming them for timing optimization based on clock periods. As illustrated in **Fig. 3**, since the clockless modules are connected with clocked ones, they regard the clockless modules as combinational paths between latches or flip-flops. In Fig. 3, rounded rectangles exemplify sub-modules in the clockless interconnect, and arrows exemplify combinational loop

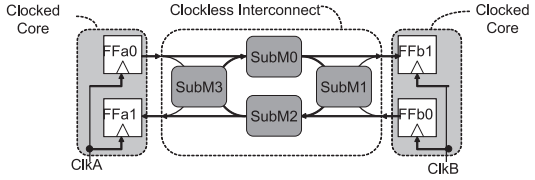

Fig. 3 Paths to be excepted from timing checking.

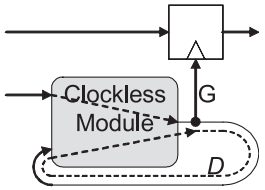

Fig. 4 Pulse generating paths possibly violating minimum pulse width constraint.

paths through the sub-modules. For example, FFa0 and FFa1 are located in a clock domain with ClkA, and the interconnect has two paths from FFa0 to FFa1. One of the paths passes a sub-module SubM3, and the other passes sub-modules SubM0, SubM1 and SubM2. The synthesis tools for clocked circuits regard the paths to be optimized so that their delays become shorter than the cycle time of ClkA. Since the clocked design tools do not consider specific design constraints on clockless design, they may create circuit structures causing hazards in the clockless modules.

Second, signals supplied to the clock ports of latches and flip-flops may violate their timing constraints. In clocked design, clock signals are supplied to the ports, and their cycle times are specified to be sufficiently longer than the timing constraints on their minimum pulse width. The constraints are, therefore, not explicitly taken into consideration in clocked design. On the other hand, in clockless design, the signals supplied to the clock ports are generated from internal signals and external input signals, as illustrated in **Fig. 4**. Timings of the signal transitions, therefore, depend on those of the external input and internal signals. The interval between two signal transitions thus may be shorter than the constraints.

Finally, the delays from isochronic forks are not nearly equal after the physical

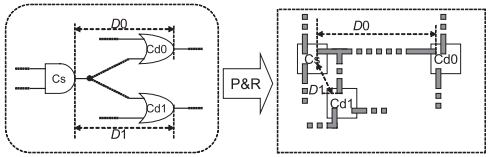

Fig. 5 Unbalanced fork by placement and routing.

synthesis, even if they are estimated to be nearly equal in the logic synthesis. This is depicted in **Fig. 5**. When the source and destinations of an isochronic fork are contained in one module, physical synthesis tools place them with timing optimization or area compaction. Then, they do not take care of equality of the delays from the fork. The circuit after physical synthesis possibly does not work correctly.

In order to overcome these difficulties, we make design settings and constraints for clockless logic, as will be explained along with the entire design flow in Section 3.2 through Section 3.4.

3.2 RTL Design

A GALS system of clocked and clockless modules is designed at RTL and netlist level (NL), respectively. Clocked modules can be designed in Verilog, and the description is synthesized by commercial logic synthesis tools. On the other hand, although description languages for clockless modules have been proposed [13),14)], no clockless logic synthesis tools have been accepted yet as widely as clocked ones. Since our proposal is to establish a design flow based on clocked design tools, the clockless modules are, hence, designed not at RTL but at NL here.

Although the clocked modules in the GALS system can be designed in the same way as in GS design, it is necessary to design carefully their connection with the clockless modules.

When the clocked modules in the GALS system are designed, the borders with the clockless modules have to be composed of flip-flops. In general, asynchronous signals input to clocked circuits have to be synchronized, and during the synchronization, logic operations to the input signals are avoided. When synchronous signals are output from clocked circuits with complicated logic operations, they may have glitches or short-period pulses. Such glitches or pulses may lead clockless circuits to malfunction. The signals, therefore, have to be received by flip-flops before they are output to the clockless modules.

3.3 Logic Synthesis

The RTL/NL mixed design of the GALS system is converted completely into a netlist by logic synthesis tools for clocked circuit design. Then, we imposed the following design constraints on the logic synthesis:

C-1: Clock cycle periods are specified for the clocked modules in the GALS design.

C-2: All paths through the clockless modules are specified as false paths.

C-3: Minimum delays are specified for signal paths for generating pulses to drive latches or flip-flops.

Since the GALS design contains the clocked modules, the clock cycle periods are specified in the same manner as the GS design, as described in C-1.

On the other hand, all the paths through clockless modules are set as false paths, as described in C-2, for solving the difficulty pointed out by Fig. 3. Since the delays of the paths through the clockless modules are allowed to be larger than any of the clock cycle periods, timing constraints based on clock cycles do not have to be imposed. The logic synthesis tools, however, regard the paths through the clockless modules as combinational paths between latches and flip-flops. Then, they try to transform and optimize the clockless modules. Consequently, this optimization possibly destroys their functionality, creating circuit structure causing hazards. The paths should be, therefore, excepted from the timing optimization.

Even for the clockless modules, there is, however, one timing constraint that has to be taken into consideration. It is the minimum path delay constraints on signal paths generating pulses to drive latches or flip-flops, as described in C-3.

The minimum path delay constraints are specified in the clockless module design to guarantee that the latches and flip-flops work correctly.

In general circuit design, for correct behavior of latches and flip-flops, the following types of timing constraints on signal arrival at the data and clock input ports of a latch or flip-flop have to be met:

- Setup time constraints.
- Hold time constraints.

- Minimum pulse width constraints.

Setup or hold time is the minimum interval between signal transition arrival times at the clock and data input ports for correct behavior of the latch or flip-flop. The minimum pulse width is the minimum interval between rising and falling edges of the clock signal that guarantees the latch or flip-flop correctly works.

Among these types, the setup and hold constraints have to be taken into consideration in general circuit design. The minimum pulse width constraints are not explicitly specified in clocked design, since they are met with specification of the clock frequency.

In clockless circuit design, however, it is not guaranteed that the minimum pulse width constraints are met. As illustrated in Fig. 4, a clockless circuit generates pulses driving internal latches or flip-flops from its internal signals and external input signals. In this figure, for simple explanation, signal G controlling the latch or flip-flop is directly fed back to an input port of the circuit. The circuit rises up or falls down the signal G, after observing G has been fallen down or risen up, respectively. Then, the interval between edges of signal G is, at shortest, the delay of the feedback loop, D. It may be thus shorter than the minimum pulse width constraints.

In order to meet the minimum pulse width constraints, they are set as the minimum path delay constraints on the feedback loop paths for the pulses. In this figure, assuming $MinPW$ is the minimum pulse width constraint value, $D > MinPW$ is set on the path generating signal G.

No other timing constraints are necessary for the logic synthesis of the GALS design due to the QDI module assumed in the design flow. It is, however, necessary to take care of isochronic forks where delays on the paths from a branch must be nearly equal. They are considered in the physical synthesis, as will be described in the next subsection.

3.4 Physical Synthesis

Physical synthesis is floorplanning, placement, clock tree synthesis (CTS) and routing, which are applied after logic synthesis. Since GALS design contains clocked cores, it is essential to synthesize clock trees for the cores.

It is important to floorplan the clockless modules in the GALS design according to QDI model, since this leads to the success in the later design steps. In the logic synthesis, the circuit structure has been decided, based on the assumption of the QDI model. There is, however, no guarantee that the estimated delays of cells in the logic synthesis are preserved after the physical synthesis. The physical synthesis thus has to be applied to the resultant netlist so that the delays fit to the assumption. Then, the key point is that although the details of paths delays are determined after routing, they are largely determined in the floorplanning.

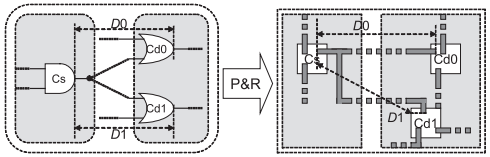

Fig. 6 Balanced fork by placement and routing.

For floorplanning of the GALS design, the clockless modules are hierarchically partitioned into sub-modules so that the following two conditions hold for each isochronic fork, as illustrated in **Fig. 6**:

- The source of each isochronic fork is contained in a different sub-module from its destinations.
- The destinations of each isochronic fork are contained in the same sub-module.

This partitioning limits the regions to place the source and destinations. No matter where the source and destinations are placed, their distances are also limited to be close to the sub-module distance. Finally, based on the floorplan, the remaining physical synthesis steps are applied to the netlist with the design constrains imposed in the logic synthesis. Consequently, the partitioning mostly makes the delays equal through the isochronic fork.

Our design flow does not need large additional efforts on the physical synthesis, compared with full synchronous design. Whereas the floorplanning requires just small additional efforts, the placement, CTS and routing are applied in the same manner.

3.5 Netlist-level Simulation

The GALS design can be simulated at netlist level for verification with delay information back-annotated from the layout information.

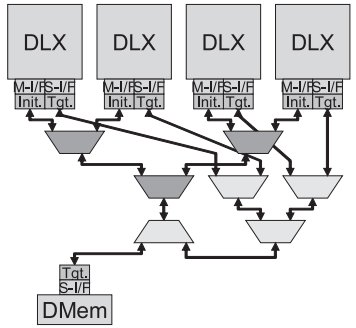

Fig. 7 Interconnect structure.

Generally, since simulators assume GS circuits in default, they assert all timing violations to terminate the simulation if any. In GALS design, such timing violations are often found at clocked latches or flip-flops receiving signals from clockless modules, such as FFa1 or FFb1 in Fig. 3. In the violations, nevertheless, GALS design keeps on working correctly. All these latches and flip-flops, hence, have to be specified as exceptions from the timing checking.

4. Case Study

In this section, we describe a case study applied to our proposed design flow which was proposed in Section 3. The design specification of the GALS four-core DLX processor [16] for the case study is illustrated in **Fig. 7**. An AHB-compliant master interface is attached to each processor core for accesses to the data memory, which has an AHB-compliant slave interface [17]. Each processor core also has an AHB-compliant slave interface to communicate with other processor cores.

The clockless interconnect consists of the following components:

- Adapters.
- Merging modules.
- Branching modules.

An adapter connects the AHB-compliant master/slave interfaces with the clockless interconnect, and is called an initiator/target. An initiator receives and packetizes AHB signals from the master interface. Packets are sent through the clockless interconnect to their destination target. The destination target depacketizes them to obtain the AHB signals, and send them to the slave interface. This access is called a command. The reverse access, i.e., from slaves to masters, is called a response.

A merging module arbitrates incoming packets to grant and output the earliest one of them. A branching module reads routing information in the header of the incoming packet to send it out for its destination. Although these modules are named after their behavior for command accesses, they also work alternatively for response access. Namely, for response accesses, a merging module reads routing information to send it out for its destination master as well. A branching module arbitrates incoming packets to send the earliest one for its destination master.

The clockless interconnect is structured like a shared bus, as illustrated in Fig. 7, compared with the clocked counterpart design. The GALS design has the following two differences from the clocked counterpart. One is the distributed arbitration by three merging modules. The other is implementation of narrower paths for inter-processor accesses due to their lower speed requirements.

We applied the design flow for 90 nm technology to the clocked DLX processor cores and AHB interfaces designed at RTL and the clockless interconnect obtained at netlist level.

The clockless interconnect was designed, based on QDI pipeline latches consisting of symmetric C-elements, OR gates and inverters with the 1-of-4 encoding [11),15)]. Each pipeline stage consists of four C-elements, whose outputs are connected with the C-elements in the next stage. The sequence of C-elements thus forms a data path in the interconnect. With the 1-of-4 encoding, a two-bit data with request is encoded into a four-bit one-hot signal on data paths. The request resetting phase is encoded into a four-bit all-zero signal. By taking logic OR of the C-element outputs, hence, the acknowledge signal is generated and input to each of its C-elements in the previous stage.

The initiators and targets were designed as mixture with clocked modules designed at RTL. The clocked modules include state machines to communicate with the cores in the AHB protocol. They locate encoder or decoder logic between 1-of-4 codes and the original AHB signals more closely to the clockless interconnect. In order to look up the request or acknowledge signal generated from the 1-of-4 codes, they also contains two flip-flop synchronizers. For higher throughput by

Fig. 8 Design result.

Table 1 Comparison: Area and power dissipation.

Circuits	Areas (mm^2)	Power (mW)
Clockless interconnect	0.6977	16.91
Clocked AHB	0.0103	0.39

working the synchronizers in parallel, they are equipped with FIFO buffers along with the borders between the clocked and clockless modules.

The design flow resulted successfully in the netlist and layout information, as illustrated in **Fig. 8**. In the experimental design flow, we utilized Design Compiler for the logic synthesis, SoC Encounter for the physical synthesis and ModelSim for netlist level simulation.

We also confirmed that the GALS design correctly works to prove the design flow, by simulating the netlist with delay information back-annotated from the layout information.

Table 1 shows area and power comparison between the clockless interconnect in the experimental GALS design and the clocked AHB bus. In the floorplanning, the maximum cell density was configured to be 70%.

According to Table 1, the clockless interconnect occupies 70-time larger area and 43-time more power than the clocked AHB bus. This large difference is partly

Table 2 Comparison: Throughput (MHz).

Height		3 mm	10 mm
Clockless I/C	1 Proc.	13	15
	4 Procs.	33	35
Clocked AHB		50	50

brought by initiator and target adapters that have no equivalent components in the clocked AHB bus. Breaking down the total area of the interconnect into those of modules, we found that the adapters' area occupies more than 83% of the clockless interconnect area in total. In addition, the difference also comes from the multiple-rail coding which requires multiple signal lines for representing one-bit data. Since our implementation requires two lines for one bit, the area would be mostly doubled, compared with the clocked counterpart, even if the adapters were not implemented. The large area difference also leads to the large power difference.

Although the clockless interconnect has area and power overheads in comparison with the clocked AHB bus, they are acceptably small when the interconnect is implemented with large clocked cores. The difference is just $0.69\,mm^2$, whereas there have been many chips with a large number of cores and tens of square millimeter size. It is possible to keep the number of adapters small by partitioning coarsely an on-chip system to be implemented. We will thus be able to find on-chip system implementation based on clockless interconnects with negligible area overhead.

The results of area and power also imply that clockless logic is potentially more power-efficient than clocked logic, if these are implemented within the same area. According to Table 1, the clockless interconnect dissipates 64% power per area of the clocked AHB bus. The clockless interconnect has no clock trees. Some modules in the interconnect dissipate no dynamic power while they are not requested to work. These features consequently save the power of the clockless interconnect.

Table 2 shows comparison of throughput between the clockless interconnect and the clocked AHB bus. The throughput was obtained from netlist level simulation with the netlist and delay information, where all clocked cores run at 200 MHz. It was measured as the number of AHB transactions per second, which

is denoted as "MHz" in this table for the sake of simplicity.

Since handshaking cycle time is longer as its physical area is larger for the same design, the throughput of the clockless interconnect depends on its physical area. The physical synthesis was applied to two different core sizes, whose widths are same but lengths are 3 mm and 10 mm in Fig. 8.

The throughput of the clockless interconnect also depends on the number of masters simultaneously requesting accesses, due to its distributed three arbiters, as illustrated in Fig. 7. Assume all the four masters have requested accesses at the same time. They are arbitrated by the first two merging modules. As a result, the modules grant two of the accesses. The other two accesses wait for completion of the two granted ones, but are not cancelled. The granted accesses are also arbitrated by the other merging module. The non-granted access of them waits for completion of the granted one, but is not cancelled. After the completion of the finally granted accesses, the three non-granted accesses are restarted in the merging modules. The clockless interconnect is thus capable of partly parallel accesses. This makes the throughput for multiple master accesses higher than that for single master accesses.

As shown in Table 2, the throughput of the clockless interconnect is thus lower than that of the clocked AHB for any number of masters running and any length of core region. The reasons are the needs of two-time packetization, synchronization and depacketization for a round trip, which result in long latency. As defined in the specification [17], when an AHB master makes two transfers, it has to wait for the response for the command of the first one, before starting the second one. In other words, each transfer needs a round trip, which performs packetization, synchronization and depacketization twice each. For instance, since two flip-flop synchronizers were adopted, a transfer takes at least four cycles longer, in addition to clocked AHB transfer latency with packetization and depacketization. This long latency results in the lower throughput.

The throughput of the clockless interconnect can be enhanced by adopting another bus protocol for communication between clocked bus interfaces and adapters. The lower throughput in the GALS design is partly brought by the protocol where a master cannot start a new transfer until receiving the response of the previous transfer. Namely, if a master can send multiple commands to slaves without waiting for responses, high throughput, possibly one transfer for every cycle, can be achieved. Then, the clockless interconnect will be able to provide a good trade-off among area, power and throughput.

5. Summary and Conclusions

In this paper, we presented a case study of a GALS design which is a mixture of clocked cores and a clockless QDI model interconnect. In particular, we were focused on how to apply design tools for clocked circuits to the clockless modules in the GALS design, and established a design flow based on clocked design tools. We also applied the design flow to an experimental GALS design containing a clockless interconnect. We proved with the experimental design that our design flow works successfully. Since the experimental design was a preliminary version, the area, power dissipation and throughput were still to be improved. Even though it may be still difficult for clockless interconnects to take over clocked buses by themselves, GALS architecture has great potential to reduce power of the entire system. In GALS architecture, the clocks supplied to the on-chip cores can be arbitrarily configured, regardless of synchronicity among them. By setting their frequencies to be minimum for the running programs, their circuits can be slowed down so as to run with lower supply voltages. It is thus clear that the GALS architecture provides power reduction. With the frequency and voltage control mechanism, the proposed design flow will help realization of lower-power GALS systems.

References

1) Chapiro, D.M.: Globally-asynchronous locally-synchronous systems, Ph.D dissertation, Computer Science Dept., Stanford Univ. (1984).
2) Zhiyi, Y. and Baas, B.M.: Performance and power analysis of globally asynchronous locally synchronous multiprocessor systems, *Proc. Int'l Symposium on Emerging VLSI Technologies and Architectures* (Mar. 2006).
3) Bormann, D.S. and Chueng, P.Y.: Asynchronous wrapper for heterogeneous systems, *Proc. Int'l Conf. Computer Design*, pp.307–314 (Oct. 1997).
4) Chattopadhy, A. and Zilic, Z.: GALDS: A complete framework for designing multiclock ASICs and SoCs, *IEEE Trans. on VLSI Systems*, Vol.13, Issue6, pp.641–654 (June 2005).
5) Gurkaynak, F.K., Oetiker, S., Kaeslin, H., Felber, N. and Fichtner, W.: GALS at

ETH Zurich: success or failure?, *Proc. Int'l Symposium on Asynchronous Circuits and Systems* (Mar. 2006).
6) Bink, A. and York, R.: The beauty of designing clockless LSIs, *Nikkei Electronics*, pp.137–146 (Sept. 2006). (in Japanese)
7) Kondratyev, A. and Lwin, K.: Design of Asynchronous Circuits Using Synchronous CAD Tools, *IEEE Design & Test of Computers*, Vol.19, Issue4, pp.107–117 (July-Aug. 2002).
8) Ozdag, R.O. and Beerel, P.A.: An Asynchronous Low-Power High-Performance Sequential Decoder Implemented With QDI Templates, *IEEE Trans. on VLSI Systems*, Vol.14, No.8 (Sep. 2006).
9) Lines, A.: Asynchronous interconnect for synchronous SoC design, *IEEE Micro*, Vol.24, Issue1, pp.32–41 (Jan. 2004).
10) Martin, A.J.: The limitation to delay-insensitivity in asynchronous circuits, *Proc. 6th MIT Conf. Advanced Research in VLSI*, pp.263–278, MIT Press (1990).
11) Bainbridge, W.J. and Furber, S.B.: Delay-Insensitive System-on-Chip Interconnect Using 1-of-4 Data Encoding, *Proc. 7th Int'l Symposium on Asynchronous Circuits and Systems* (*ASYNC 01*), pp.118–126 (2001).
12) Udding, J.T.: A formal model for defining and classifying delay insensitive circuits and systems, *Distributed Comput.*, Vol.1, pp.197–204 (1986).
13) van Berkel, C.H., Kessels, J., Roncken, M., Saeijs, R., and Schalij, F.: The VLSI-programming language Tangram and its translation into handshake circuits, *European Conf. on Design Automation*, pp.384–389 (1991).
14) Berdsley, A. and Edwards, D.A.: The Balsa asynchronous circuit synthesis system, *Forum on Design Language* (Sep. 2000).
15) Sparsø, J. and Furber, S.: *Principles of Asynchronous Circuit Design*, pp.9–28 and 57–80, Kluwer Academic Publishers (Dec. 2001).
16) Hennessy, J.L. and Patterson, D.A.: *Computer architecture: A quantitative approach*, second edition, Morgan Kaufmann Publishers (Jan. 1996).
17) ARM: AMBA Specification (Rev.2.0), pp.3-1–3-58 (1999).
18) Tanaka, K., Nakamura, Y. and Atarashi, A.: A Case Study of Multi-processor Design with Asynchronous Interconnect using Synchronous Design Tools, *Proc. 14th workshop on Synthesis and System Integration on Mixed Information Technologies* (*SASIMI 2007*), pp.287–293 (Oct. 2007).

(Received December 25, 2007)
(Revised March 17, 2008)
(Accepted May 1, 2008)
(Released August 27, 2008)

(Recommended by Associate Editor: *Hiroshi Saito*)

Katsunori Tanaka received his Ph.D. degree from Kyoto University in 2005. He had been a visiting scholar at the University of Illinois at Urbana-Champaign from 2000 to 2002. He is currently a research staff member at System IP Core Research Labs., NEC Corp. His research interests include combinational logic synthesis and asynchronous logic design methodology.

Yuichi Nakamura received his B.E. degree in information engineering and M.E. degree in electrical engineering from the Tokyo Institute of Technology in 1986 and 1988, respectively. He received his D.E. degree from the Graduate School of Information, Production and Systems, Waseda University, in 2007. He is currently a senior principal researcher at System IP Core Research Labs., NEC Corp. His research interests include the system LSI design and their algorithms.

Atsushi Atarashi received his B.E. degree in information engineering and M.E. degree in electrical engineering from the Tokyo Institute of Technology in 1983 and 1985, respectively. He is currently a senior manager at System IP Core Research Labs., NEC Corp. His research interests include multimeda oriented processor architecture and software.