Hindawi Publishing Corporation
VLSI Design
Volume 2014, Article ID 531653, 10 pages
http://dx.doi.org/10.1155/2014/531653

Research Article

Design of Smart Power-Saving Architecture for Network on Chip

Trong-Yen Lee and Chi-Han Huang

Department of Electronic Engineering, National Taipei University of Technology, Taipei 10608, Taiwan

Correspondence should be addressed to Trong-Yen Lee; tylee@ntut.edu.tw

Received 5 June 2014; Accepted 27 June 2014; Published 6 August 2014

Academic Editor: Yu-Cheng Fan

In network-on-chip (NoC), the data transferring by virtual channels can avoid the issue of data loss and deadlock. Many virtual channels on one input or output port in router are included. However, the router includes five I/O ports, and then the power issue is very important in virtual channels. In this paper, a novel architecture, namely, Smart Power-Saving (SPS), for low power consumption and low area in virtual channels of NoC is proposed. The SPS architecture can accord different environmental factors to dynamically save power and optimization area in NoC. Comparison with related works, the new proposed method reduces 37.31%, 45.79%, and 19.26% on power consumption and reduces 49.4%, 25.5% and 14.4% on area, respectively.

1. Introduction

In recent years, the 3-dimensional IC and TSV (Through-Silicon Via) technology are proposed to solve area issues. The 3-dimensional IC of Intel Ivy Bridge processor and the 16-core multicore architecture can be implemented in 22 nm [1]. Therefore, the multicore and heterogeneous systems are popular research in SoC (system-on-chip). These architectures require high throughput and performance to transfer data in a multicore SoC. Therefore, the NoC (network-on-chip) can be proposed to solve this requirement, but it derived new problems such as power consumption and area [2, 3].

The NoC architecture [1] consists of processing element (PE), network interface (NI), router, and topology which is shown in Figure 1. The PEs transfer information to NI, the NI packages the information into flits then passes to routers. The routers have difference corner router (CR), edge router (ER), and router (R); the CR, ER and R has three, four, and five I/O ports to access information then each port includes n virtual channels. Router includes transmission channel, routing computation (RC), virtual channel arbiter (VA), switch arbiter (SA), and crossbar (XBAR). The flits includes header, body, and tail; the header flit has PE priority, source address, destination address, and so forth. The RC uses header flit and routing algorithms to find transmission path. VA uses two stages arbitration to select most high priority packet transmission and then will sign transmission channel. SA uses two stages arbitration and will select most body flits into XBAR to transmit. The VA will be working when the packet is arrival. The SA operation when the flit is arrival. The tail flit represents last flit, and then the router will unregister transmission channel. The router topology includes mesh, star, and fat tree [4, 5].

Yoon et al. [6] analysis of virtual channels (VCs) can avoid routing and protocol deadlock and improve the routing performance when the packet traffic is congested. The VCs can solve packet switch hard issue but it leads the power and area and so forth issue in NoC.

Nicopoulos et al. [2] proposed IntelliBuffer architecture to solve PV (process variation) to reduce the power consumption in layer 1 [7]. It differs from the conventional architecture in two fundamental ways. First, these slots use clock-gating to reduce the power consumption when slots are empty. In order to avoid data loss transmission, one of slots clock keeps to access data in each I/O port. Second, the router creates a leakage classification register (LCR) table; then the write and read pointer always accesses the lowest power consumption slots from the LCR table.

Taassori et al. [3] proposed an adaptive data compression technology to reduce the number of packet bits in layer 3 [7]. It reduces of the number of transmissions. Therefore, it can improve power consumption of router. Palma et al. [8] use T-Bus-Invert technology to reduce the hamming distance transition activity rate to improve the power consumption.

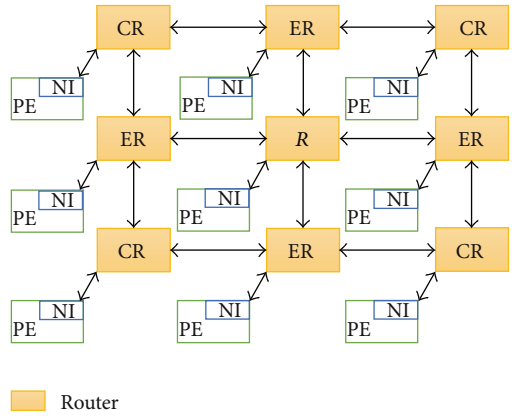

Figure 1: NoC architecture.

Jafarzadeh et al. [9] use end-to-end data coding technology to minimize switching activity rate and routing path to improve NI power consumption.

Lee et al. [10] proposed buffer clock-gating architecture and used clock-gating to reduce the transmit power consumption when slots are empty and full. Ezz-Eldin et al. [11] proposed an adaptive virtual channel with two sections in layer 1 [7]. First, the work used hierarchical multiplexing tree for Virtual Channels (VCs) to reduce area. Second, it uses clock-gating to reduce power consumption. Rosa et al. [12] proposed dynamic frequency scaling in PE for NoC. It considers the communication and loading rate to control the router frequency to reduce the power consumption.

Huaxi et al. [13] proposed fat tree-based optical NoC; this architecture includes topology, placement, layout, and protocol. This paper proposed low power and cost router optical turnaround router to improve the power consumption. Gu et al. [14] proposed Cygnus router to optimize the router algorithms to reduce the power consumption. Swaminathan et al. [15] create two FIFOs in NI. Use two FIFO dynamic configuration data access to improve throughput and power consumption.

In the next section we analyse the power consumption under the difference VCs access. Section 3 we introduce the topology and router packet architecture, we addition the SPS in router to save power. In Section 4 we present SPS with router design. Section 5 contains experimental results and Section 6 concludes this paper.

2. Power Issue with Virtual Channels

The multicore architecture and big data communication are more popular in next generation. Traditional communication technologies cannot meet a large amount of traffic on multicore and heterogeneous chip. The NoC can solve this issue. It uses network transmission method to make the difference core communication at same time. The NoC can solve the communication issue but the big data access enhances the power consumption.

The router composed of the arbitration and transmission unit [16] is illustrated in Figure 2. The arbitration unit selects the highest priority packet sent to next router. The arbitration unit includes routing computation (RC), VC arbiter (VA), and switch arbiter (SA). The RC is the calculation of routing paths and priorities. The VA contains a number of two-stage arbitrations to select packet and sign up VCs. First stage selects the local highest priority packet from input VCs to crossbar and signs up VCs. Second stage selects the global highest priority packet from input crossbar to output VCs and signs up VCs. The SA also contains a number of two-stage arbitrations to select flits for transmission. First stage selects the local highest priority flits from input VCs to crossbar. Second stage selects the global highest priority flits from input crossbar to output VCs. The VA executed prepacket and the SA executed preflits.

The router with transmission unit is illustrated in Figure 3. In this unit, it includes nVCs to access large packet from input physical channel to output physical channel. A power consumption calculation to VCs is shown in (1). The variable of n represents the number of access packets or flits in VCs. The variable of f represents access frequency in VCs. The variable of c represents capacitance and v represents voltage in VCs. Nicopoulos et al. [2] and Katabami et al. [17] proposed clock-gating to solve this issue.

In this paper, we proposed a dynamic control of each virtual channel clock in different transmission environments. Whether packet transfer is complete, the SPS can effectively reduce the power consumption and does not affect the transmission performance. Consider

$$P_{n\mathrm{VCs}} = \sum_{n=1}^{\infty} 1^n \times \left(f \times c \times v\right)^2. \tag{1}$$

3. Router and Topology with SPS

3.1. Relation of Topology and Router. The relation of topology and router is illustrated in Figure 4. The router uses different transmission mode with topologies. For example, the mesh uses the X-Y routing to transmit. The X-Y routing flow chart for 2×2 meshes is illustrated in Figure 5, when the MSB of destination router address (R_{dm}) is equal to the MSB of current router address (R_{cm}) and if the LSB of router

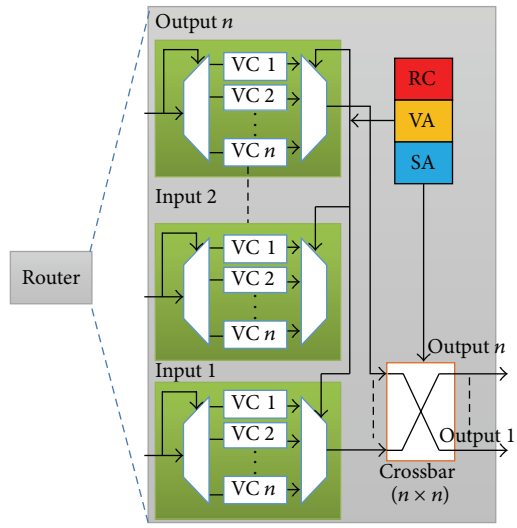

Figure 2: Router architecture with NoC.

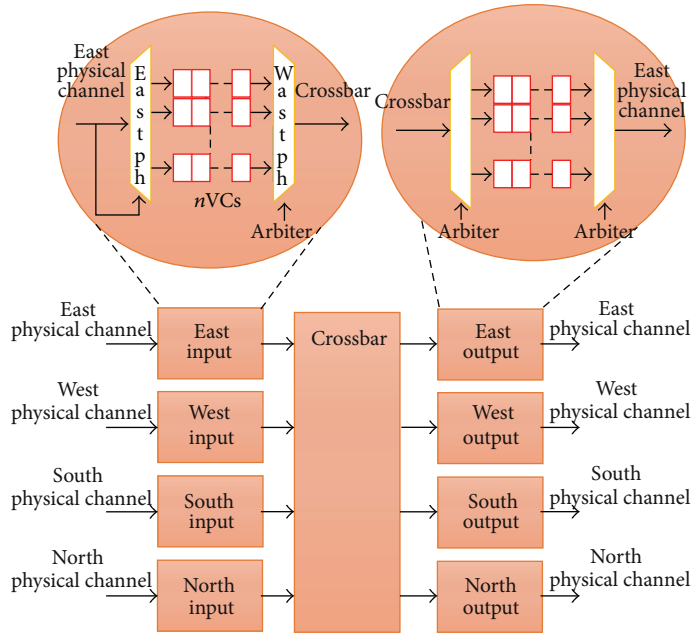

Figure 3: Transmission unit.

addresses (R_{dl} and R_{cl}) is equal then it means the flits arrival. Otherwise, the X-Y routing algorithm includes two-stage flows. In stage one, the flits are sent until that the R_{dm} equals of R_{cm} on the x-axis routers. In stage two, the flits are sent to the destination by y-axis routers. The virtual channel will be initialed under packet transmit on two routers, which procedure is shown on Algorithm 1.

The control method of arbiter architecture uses different transmission mode to design. The VC arbiter and switch bar are by the topology and priority to design the routing computation unit. Algorithm 2 constructs VC two stages arbitration of prepackets. Stage 1 decided high priority packet into crossbar from local VCs (input VCs) of each packet at lines 3 to 4 and lines 8 to 10. Stage 2 decided most important packet to transmission from global VCs (output VCs) of each packet at lines 5 to 6 and lines 11 to 13.

Sign up Algorithm
Input: R_{roth} and E_{mp}.
(1) while (flits arrival) do
(2) if (R_{rothf2} is header and adx is free channel)
(3) {sign up the channel and select the channel to output}
(4) else if (R_{rothf2} is body and $adx = R_{roths2}$)
(5) {select the channel to output}
(6) else if (R_{rothf2} is tail and $adx = R_{roths2}$)
(7) {clear the channel and select the channel to output;}
(8) else
(9) {read back flit to virtual channel}
(10) end while

Algorithm 1: Channel sign up algorithm.

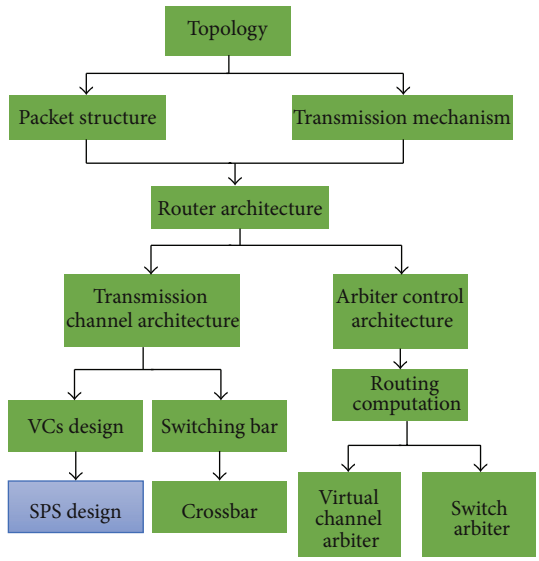

Figure 4: Topology and router relation with SPS.

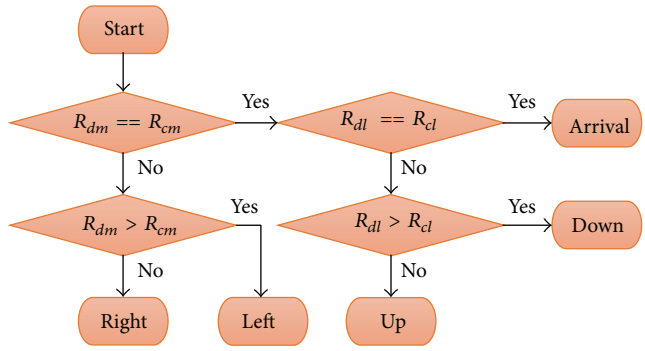

Figure 5: X-Y routing flow chart.

Virtual channel arbitration
Input: header flits
/∗Control signal enable∗/
(1) while (header flits) do
(2) use lottery arbitration to select local and global highest priority flits
(3) if (local)
(4) {V_{ai} = local input virtual channel address}
(5) if (global)
(6) {V_{ao} = global input virtual channel address}
(7) end while
/∗Channel switch∗/
(8) Case V_{ai}
(9) {C_{ri1} = local packet of V_{ai}}
(10) end case
(11) Case V_{ao}
(12) {R_{it} = global packet of V_{ao}}
(13) end case

Algorithm 2: VC arbitration algorithm.

Switch arbitration
Input: body and tail flits
/∗Control signal enable∗/
(1) while (body or tail flits) do
(2) use channel sign up register to select local and global highest priority flits
(3) if (local)
(4) {S_{ai} = local input virtual channel address}
(5) if (global)
(6) {S_{ao} = global input virtual channel address}
(7) end while
/∗Channel switch∗/
(8) Case S_{ai}
(9) {C_{ri2} = local packet of S_{ai}}
(10) end case
(11) Case S_{ao}
(12) {R_{ot} = global packet of S_{ao}}
(13) end case

Algorithm 3: Switch arbitration algorithm.

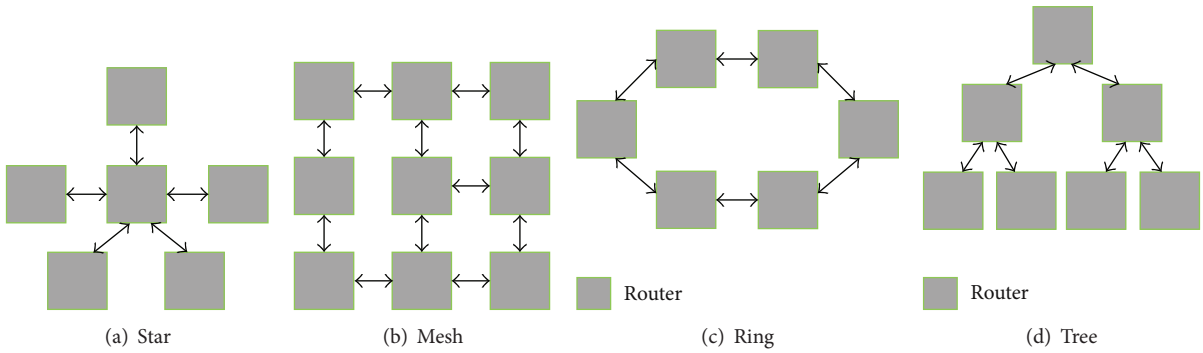

Figure 6: Router connection topology architecture.

Algorithm 3 constructs VC two stages arbitration of preflits. Stage 1 decided high priority flit into crossbar from local VCs (input VCs) of each flit at lines 3 to 4 and lines 8 to 10. Stage 2 decided most important flit to transmit from global VCs (output VCs) of each flits at lines 5 to 6 and lines 11 to 13.

The router includes four directions to connect other routers and one local physical channel to connect PE in transmission channel architecture. There have been nVCs of each physical channel without local physical channel. The switch bar support for transmission the most important packet to output channel. The SPS controls each VCs power consumption when the channel status changes. The SPS architecture is introduced in next section.

3.2. Topology Architecture. The topology is definition of the packet transmission path between router and link. The router connection topology architecture is shown in Figure 6; they include star, mesh, ring, and tree topologies. The RC algorithms depend on topology architecture in arbitration unit. The VA and SA algorithms depend on packet priority in arbitration unit. In this paper, the topology is the 2×2 mesh, the RC algorithm is X-Y routing, and the VA and SA algorithms are lottery [18].

The router that connects with PE is shown in Figure 7; so that the PE and router access information, use the network interface (NI). It handles the information between router and PE. The NI includes two level designs [19] as shown in Figure 8. It contains three modules to meet the specifications of the different layers. The shell module needs to meet IP specification. The kernel module needs to meet the NoC topology specification.

3.3. Flits with Router Architecture. The flit specification with router is shown in Figure 9; the flit type of 2-bit 00 represents the one packet; this flit type does not sign up VCs. The 2-bit 01 represents the *header* flit which includes routing information and address; this flit type always is determined in sign up channel. The 2-bit 10 represents the *body* flit which includes transmission information; this flit payload records the segment packet. The 2-bit 11 represent the *tail* as last transmission information; this flit not only records the last segment packet but also cleans the VCs.

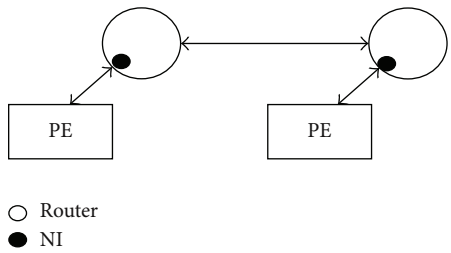

Figure 7: Router connection with PE.

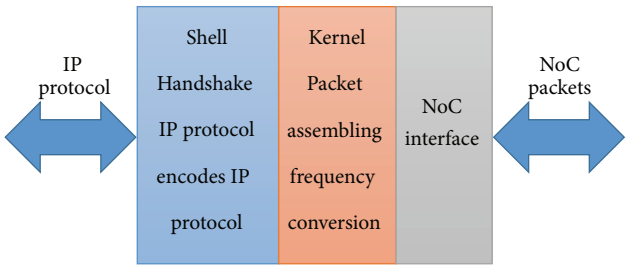

Figure 8: NI breakdown into Shell, Kernel, and interface.

Flit type (00)	Source address	Destinations address	Payload
Flit type (01)	Source address	Destinations address	Routing information
Flit type (10)	Payload		
Flit type (11)	Payload		

Figure 9: Flits type of router.

4. SPS with Router Design

The VC that contains many slots to access data led to extra power consumption. In this paper, we propose SPS architecture to reduce the power consumption.

4.1. Router with SPS Architecture. The proposed router with SPS architecture is illustrated in Figure 10. The physical channel (PC) is used to connect other routers and access information. The input VCs (IVC) is used to store information from PCs. It always is designed by FIFO or other sequential logic. The arbiter decides the flits priority to control input switch logic (ISL) and output switch logic (OSL) to transmit flits. It includes RC, VA, and SA. The crossbar (CR) connects IVC to OVC, the switch signal form arbiter. The output VCs (OVC) store information from CR. The proposed SPS uses the transmission channel status to dynamic control IVC and OVC clock in essential operating.

The VCs with SPS architecture are illustrated in Figure 11. It controls system clock into I/O VC to reduce power consumption. In this architecture, the VC contains 0 to $i-1$ slots to access data.

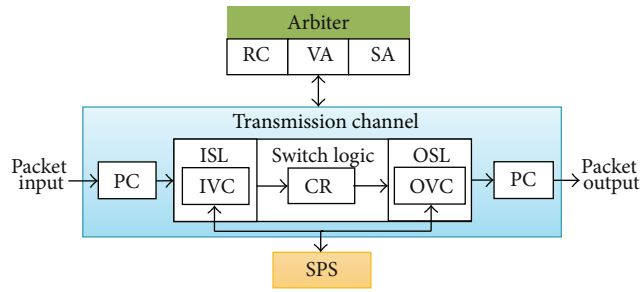

Figure 10: Router with SPS architecture.

4.2. Design of SPS Control Timimg. The VCs access timing diagrams of SPS architecture are illustrated in Figure 12. The Clock Block A indicates that the VCs have no information to transmit. The Clock Block B indicates that the VCs are writing information. The Clock Block C indicates that the data in VCs are waiting to transmit. Our analysis for unused clock-gating architecture is shown in (2). The slots access information of power consumption is denoted by P_a. The slot content full and empty of power consumption are denoted by P_f and P_e, respectively. The P_s is power consumption except for P_f, P_e, and P_a. The unused clock-gating architecture does not control clock for sequential logic in nVCs. Therefore, the logic will generate power consumption in high transmission structure.

The clocking gating consumes power in Clock Block B and Clock Block C. Our analysis for clock-gating architecture is shown in (3). The P_{g1} is power consumption of empty gating. The clock-gating architecture does not control clock when VCs is full stage. The VCs always store flits to wait for transmission.

The SPS consumes power in Clock Block B. Our analysis for SPS architecture is shown in (4). The P_{g2} is power consumption of SPS. It saves the power consumption of empty and full gating for nVCs. Consider

$$P_{r1} = P_a + P_f + P_e + P_s, \tag{2}$$

$$P_{r2} = P_a + P_f + P_s + P_{g1}, \tag{3}$$

$$P_{r3} = P_a + P_s + P_{g2}. \tag{4}$$

4.3. Design of SPS. The proposed SPS uses the VCs status to dynamic control clock of each VC. The CFSM of SPS with VCs is illustrated in Figure 13; it contains two CFSM in this architecture.

The first CFSM includes initial, empty, full, and waiting status. *Initial status*: when the VC is reset, the structure is into the initial status until the flit arrive. *Empty status*: when the user resets the VCs or the flits transport to next storage unit, the structure is into this status. *Full status*: the store flit in VC is full. *Waiting status*: When the user resest the VCs or the store flit is complete.

The VCs with SPS algorithm is illustrated in Algorithm 4. In line 3, the VCs will initialize the VCs count and flags. The VCs will access flits to change VCs count when channel packet or arbiter signal arrive at line 4 to 9. When the VCs

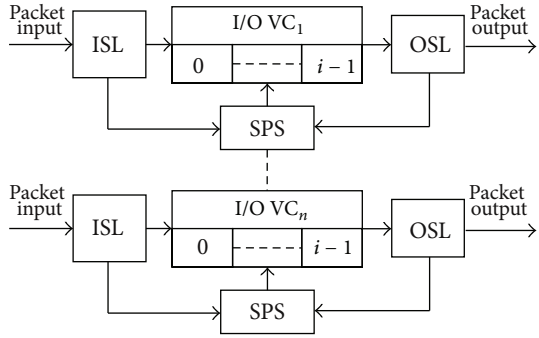

Figure 11: VCs with SPS architecture.

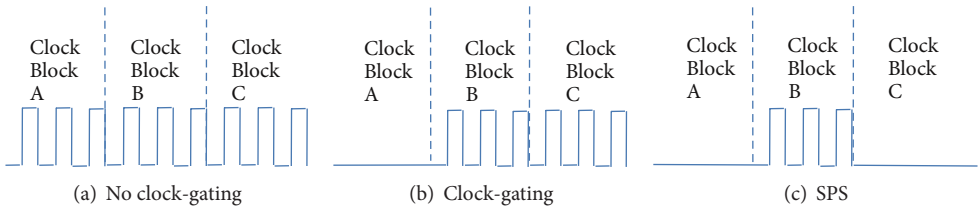

Figure 12: VCs power with clock diagram.

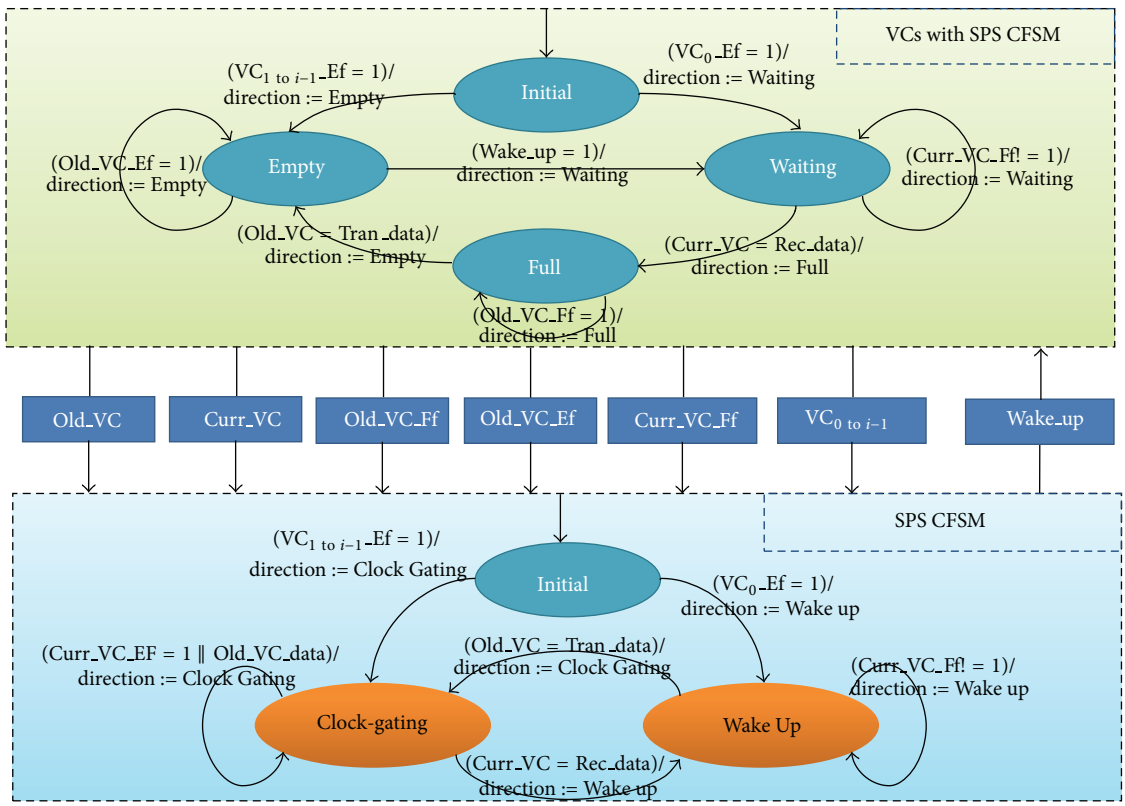

Figure 13: CFSM of SPS with VCs.

count can be changed, then the VCs flag will be changed at line 10 to 17.

The second CFSM includes initial, clock-gating, and wake up status. *Initial status*: this principle is the first CFSM of initial state. *Clock-gating*: when the VC changes to full or empty, then SPS will disable this VC clock and change to this status. *Wake up*: when the VC want to store flit, one VC will wake up.

The SPS algorithm is illustrated in Algorithm 5. In line 3, the SPS will initialize VCs clock and access status from VCs

VCs with SPS Algorithm
Input: VCs clock, channel packet, arbiter signal and reset.
Output: channel packet, channel status
(1) VC_{count} is integer and range is $1 \leq VC_{count} \leq n$
(2) VC_{flag} includes full flag and empty flag
(3) initial VC_{count} and VC_{flag}
(4) while (channel packet or arbiter signal be arrival) do
(5) if (channel packet be arrival and full flag != 1)
(6) {$VC_{count} = VC_{count} + 1$ and packet store in VCs}
(7) if (arbiter signal be arrival and empty flag != 1)
(8) {$VC_{count} = VC_{count} - 1$ and packet be read from VCs}
(9) end while
(10) while (VC_{count} be change) do
(11) if ($VC_{count} = n$)
(12) {assign full flag to 1}
(13) else if ($VC_{count} = 1$)
(14) {assign empty flag to 1}
(15) else
(16) {assign full flag and empty flag to 0}
(17) end while

Algorithm 4: VCs with SPS algorithm.

SPS Algorithm
Input: system clock, channel packet, arbiter signal and reset.
Output: VCs clock
(1) VC_{group} is VCs group of 4 direction port
(2) VC_{flag} includes full flag and empty flag
(3) Initial VCs clock and access VCs count and stage flag
(4) follow LCR to arrangement all slots priority;
(5) VC_{clki} is VCs clock of each VC_{group} //where $1 \leq i \leq n$
(6) Example VC_{group} = East port
(7) initial $VC_{clki} = 0$; //where $1 \leq i \leq n$
(8) while (virtual channel be write) do
(9) if (VC_{flag} = empty)
(10) {VC_{clki} = system clock}
(11) If (VC_{flag} = full flag)
(12) {$VC_{clki} = 0$ and VC_{clki+1} = system clock}
(13) end while
(14) while (virtual channel be read) do
(15) if (empty flag = 1)
(16) {$VC_{clki} = 0$}
(17) end while

Algorithm 5: SPS algorithm.

with VC flags. The slots priority from LCR [2] and each VCs clock can be initialized at lines 4, 5, and 7. The SPS controls VCs clock to reduce the VCs power consumption when VCs is accessed and flags changed at lines 8 to 17.

5. Experimental Results

In this section, we proposed autotesting architect for router with SPS. This architect includes four modules of autotesting. The first module is test-vector generator (TVG); the FSM is illustrated in Figure 14; the Idle status is waiting for the requirement of start testing, when the requirement arrives, TVG then will change status from idle to generator. When the requirement is cancelled, the status be changed from generator to idle. The generator status will generate test-vector and compare-vector; this is illustrated in Figure 15; we use *c* language to generate lottery arbitration [18] in test-vector at control step 1. We use HDL to design the conventional router to generate the compare-vector and the input pattern from the test-vector at control step 2. When the compare-vector and test-vector functions are complete then the status will be changed from generator to vector output (VO) at control step 3. The VO status will transform test-vector and compare-vector to Xilinx memory IP files, through memory to control data output to test and compare only one clock.

Router with SPS Algorithm
Input: system clock, start, Lottery Input.
Output: test-start, Implement-results
(1) If start testing
(2) {test-start = 1; pass VD}
(3) While (read test data from and start bit set-up to one) do
(4) Lottery Input = Test-vector
(5) Implement-results = Test-vector use Router with SPS to transmission;
(6) Test-vector address = Test-vector address + 1;
(7) If (test finish or start = 0)
(8) {test-start = 0}
(9) End while

Algorithm 6: Router with SPS testing algorithm.

The second module is vector database (VD); the control flow graph is illustrated in Figure 16; the module writes VO status vector in memory. The database includes two vectors to test and analyze the proposed circuit. The lottery database is provided test packet for router with SPS. The compare database is provided analysis for router with SPS.

The third module is router with SPS; we use VD to propose the test-vector to implement this module. The testing algorithm is illustrated in Algorithm 6, when the start signal set up to one from I/O, then the module starts to test and pass this signal to VD at lines 1 to 2. When testing is started, the input signal will be read from VD, shown at lines 3 and 4 in Algorithm 6. The read test-vector delay time is one clock from VD to router with SPS. The router with SPS uses VD test-vector to compute at line 6. When this pattern computation is finish, the next pattern will be read from VD at line 6. When the test pattern computation is finished or start signal is cancelled, test-start set up and stop testing at lines 7 and 8.

The final module, verification module, is illustrated in Figure 17; we verify the function in this module. The function verification is comparing of compare-vector and implement-results from VD and router with SPS. If the pattern is error, then verification result returns error signal.

The hardware experimental environment uses Xilinx FPGA xc5vlx50t-1ff1136 to verify SPS architecture. The software experimental environment uses Xilinx ISE 12.3 and the

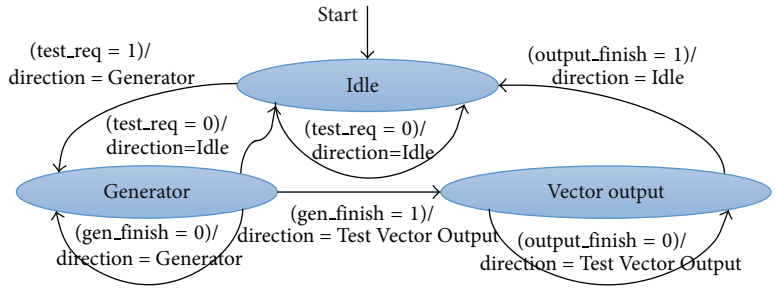

Figure 14: Test-vector generator (TVG) module FSM.

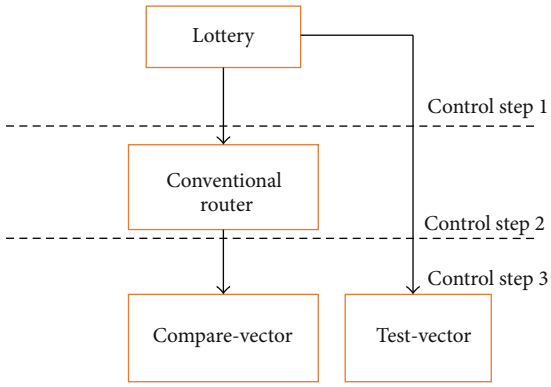

Figure 15: Generator status control and data flow graphs.

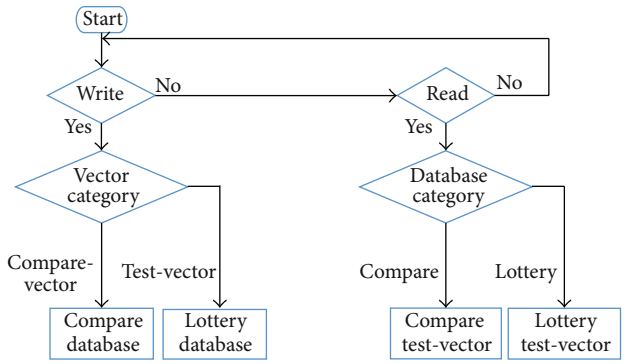

Figure 16: Vector database (VD) control flow graph.

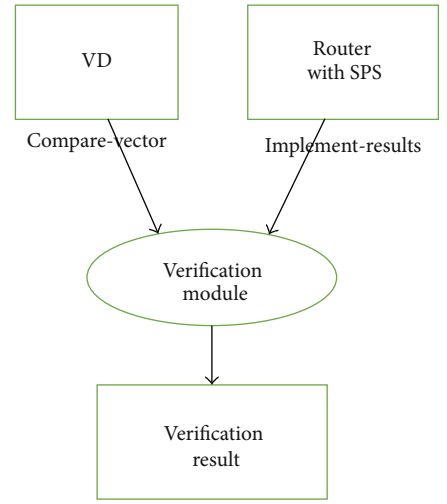

Figure 17: Verification module data flow graphs.

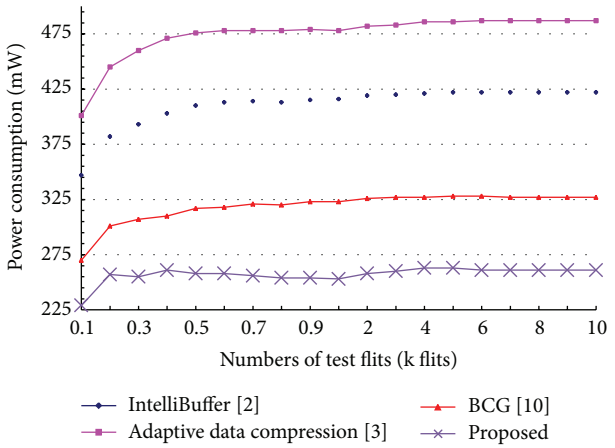

Figure 18: Power consumption distribution.

analysis tools use Modelsim 6.6, Xilinx Chipscope ILA, and Xpower 12.3, which are supported by Xilinx. The test experimental environment uses 2×2 mesh and X-Y routing; the PC have 4 VCs to access flits. The power consumption distribution is illustrated in Figure 18; the number of test packets is from 100 to 10000. The packet format is flit and packet length is 18 bits.

Comparing related works, as shown in Table 1, IntelliBuffer [2], adaptive data compression [3], and buffer clock-gating [10], the proposed method reduces 37.31%, 45.79%, and 19.26% on power consumption, respectively, and reduces 49.4%, 25.5% and 14.4% on area, respectively.

6. Conclusions

The *Smart Power-Saving (SPS)* architecture for network-on-chip was presented. A clock control circuit and SPS algorithm are demonstrated to reduce the power consumption on the NoC architecture. From experimental results, the proposed

Table 1: Comparison of power consumption and area.

Methods	Constraints			
	Power consumption (mW)	Area (number of slices)	Improved power	Improved area
IntelliBuffer [2]	410.42	1551	37.31%	49.4%
Adaptive data compression [3]	474.53	1054	45.79%	25.5%
Buffer clock-gating [10]	318.63	917	19.26%	14.4%
Newly proposed	257.05	785		

SPS architecture is more efficient to reduce the power consumption than IntelliBuffer [1], adaptive data compression [3], and buffer clock-gating [10] in the NoC architecture.

Conflict of Interests

The authors declare that there is no conflict of interests regarding the publication of this paper.

Acknowledgment

The authors would like to thank the Ministry of Science and Technology of the Republic of China, Taiwan, for partially supporting this research.

References

[1] D. James, "Intel Ivy Bridge unveiled—the first commercial tri-gate, high-k, metal-gate CPU," in *Proceedings of the Custom Integrated Circuits Conference (CICC '12)*, pp. 9–12, September 2012.

[2] C. Nicopoulos, S. Srinivasan, A. Yanamandra et al., "On the effects of process variation in network-on-chip architectures," *IEEE Transactions on Dependable and Secure Computing*, vol. 7, no. 3, pp. 240–254, 2010.

[3] M. Taassori, M. Taassori, and M. Mossavi, "Adaptive data compression in NoC architectures for power optimization," *International Review on Computers and Software*, vol. 5, no. 5, pp. 540–547, 2010.

[4] D. Bertozzi and L. Benini, "Xpipes: a network-on-chip architecture for gigascale systems-on-chip," *IEEE Circuits and Systems Magazine*, vol. 4, no. 2, pp. 18–31, 2004.

[5] S. J. Lee, K. Lee, and H. J. Yoo, "Analysis and implementation of practical, cost-effective networks on chips," *IEEE Design and Test of Computers*, vol. 22, no. 5, pp. 422–433, 2005.

[6] Y. J. Yoon, N. Concer, M. Petracca, and L. Carloni, "Virtual channels versus multiple physical networks: a comparative analysis," in *Proceedings of the 47th ACM/IEEE Design Automation Conference (DAC '10)*, pp. 162–165, June 2010.

[7] L. Benini and G. de Micheli, "Networks on chips: a new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70–78, 2002.

[8] J. C. S. Palma, L. S. Indrusiak, F. G. Moraes, R. Reis, and M. Glesner, "Reducing the power consumption in networks-on-chip through data coding schemes," in *Proceedings of the 14th IEEE International Conference on Electronics, Circuits and Systems (ICECS '07)*, pp. 1007–1010, December 2007.

[9] N. Jafarzadeh, M. Palesi, A. Khademzadeh, and A. Afzali-Kusha, "Data Encoding Techniques for Reducing Energy Consumption in Network-on-Chip," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 22, no. 3, pp. 675–685, 2014.

[10] T. Y. Lee, C. H. Huang, and X. S. Lin, "Design of buffer clock-gating architecture for network-on-chip," in *Proceedings of the 22th VLSI Design/CAD Symposium*, pp. 2–5, August 2011.

[11] R. Ezz-Eldin, M. A. El-Moursy, and A. M. Refaat, "Low leakage power NoC switch using AVC," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '12)*, pp. 2549–2552, Seoul, Republic of Korea, May 2012.

[12] T. R. da Rosa, V. Larrea, N. Calazans, and F. G. Moraes, "Power consumption reduction in MPSoCs through DFS," in *Proceedings of the 25th Symposium on Integrated Circuits and Systems Design (SBCCI '12)*, pp. 1–6, 2012.

[13] G. Huaxi, X. Jiang, and Z. Wei, "A low-power fat tree-based optical network-on-chip for multiprocessor system-on-chip," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition (DATE '09)*, pp. 3–8, April 2009.

[14] H. Gu, K. H. Mo, J. Xu, and W. Zhang, "A low-power low-cost optical router for optical networks-on-chip in multiprocessor systems-on-chip," in *Proceedings of the IEEE Computer Society Annual Symposium on VLSI (ISVLSI '09)*, pp. 19–24, Tampa, Fla, USA, May 2009.

[15] K. Swaminathan, G. Lakshminarayanan, F. Lang, M. Fahmi, and S. B. Ko, "Design of a low power network interface for Network on chip," in *Proceedings of the 26th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE '13)*, pp. 1–4, May 2013.

[16] R. Mullins, A. West, and S. Moore, "Low-latency virtual-channel routers for on-chip networks," in *Proceedings of the 31st Annual International Symposium on Computer Architecture (ISCA '04)*, pp. 188–197, 2004.

[17] H. Katabami, H. Saito, and T. Yoneda, "Design of a GALS-NoC using soft-cores on FPGAs," in *Proceeding of the Embedded Multicore Socs (MCSoC '13)*, pp. 26–28, September 2013.

[18] J. Wang, Y. Li, Q. Peng, and T. Tan, "A dynamic priority arbiter for network-on-chip," in *Proceedings of the IEEE International Symposium on Industrial Embedded Systems (SIES '09)*, pp. 253–256, July 2009.

[19] S. Saponara, L. Fanucci, and M. Coppola, "Design and coverage-driven verification of a novel network-interface IP macrocell for network-on-chip interconnects," *Journal of Microprocessors and Microsystems*, vol. 35, no. 6, pp. 579–592, 2011.