A full asynchronous serial transmission converter for network-on-chips*

Yang Yintang(杨银堂), Guan Xuguang(管旭光)†, Zhou Duan(周端), and Zhu Zhangming(朱樟明)

(Institute of Microelectronics, Xidian University, Xi'an 710071, China)

Abstract: Large transmission power consumptions and excessive interconnection lines are two shortcomings which exist in conventional network-on-chips. To improve performance in these areas, this paper proposes a full asynchronous serial transmission converter for network-on-chips. By grouping the parallel data between routers into smaller data blocks, interconnection lines between routers can be greatly reduced, which finally brings about saving of power overheads in the transmission process. Null convention logic units are used to make the circuit quasi-delay insensitive and highly robust. The proposed serial transmission converter and serial channel are implemented based on SMIC 0.18 μm standard CMOS technology. Results demonstrate that this full asynchronous serial transmission converter can save up to three quarters of the interconnection line resources and also reduce up to two-thirds of the power consumption under 32 bit data widths. The proposed full asynchronous serial transmission converter can apply to the on chip network which is sensitive to area and power.

Key words: network-on-chips; full asynchronous; serial transmission converter; threshold gate; low power
DOI: 10.1088/1674-4926/31/4/045007 **EEACC:** 1265B

1. Introduction

As semiconductor technology scales down, more IP cores are being integrated onto a single chip to implement more complicated system functions. But with the increasing working speed and larger scale of on chip systems, the conventional single clock working manner faces a lot of challenges, such as poor reusability of modules, power consumption increment of the clock tree, large size of the clock tree area, clock skew and electromagnetic interference (EMI). Because of these problems, the complexity of designing very deep submicron integrated circuit is greatly enhanced. So the problems caused by the clock have become crucial issues which need to be solved first in ultra large integrated circuits.

To reduce power consumption and increase communication performance, extensive research has been conducted into network-on-chip (NoC)[1−4] systems. The NoC approach particularly suits communication-dominant on-chip systems. Asynchronous NoCs have been proposed to eliminate the clock for global communication[5, 6], and also provide better power efficiency and higher modularity compared to synchronous NoCs. Interconnection cost, in terms of the number of wires required between routers, could grow to be considerable in NoCs if the data width is increased since each router is effectively connected by a point-to-point link to a neighboring router and the high cost of parallel links has been shown in Ref. [7], which compares fully parallel and bit-serial buffered wires. It is expected that, with further scaling down of technology, the number of point-to-point links between the switches of a NoC will grow as more and more intellectual property (IP) cores are integrated into a system. But traditional serial transmission converters[8] are applied only to quasi-synchronous network-on-chips. To the best of our knowledge, there are no related reports on serial transmission converters of full asynchronous network-on-chips.

So this paper proposes a serialization method to reduce the number of wires between NoC routers. Because full asynchronous circuits have their inherent advantages in speed and power, the full asynchronous working manner is adopted to realize the serial transmission converter. Byte-level serialization of the data is performed instead of a full bit-serial single wire link. So it can achieve a balance between performance and power consumption. By adopting null conventional logic gates, the circuit has high robustness and a characteristic of quasi-delay insensitivity. The proposed serialization converter is preferable to network-on-chips which have limited resources of area and power.

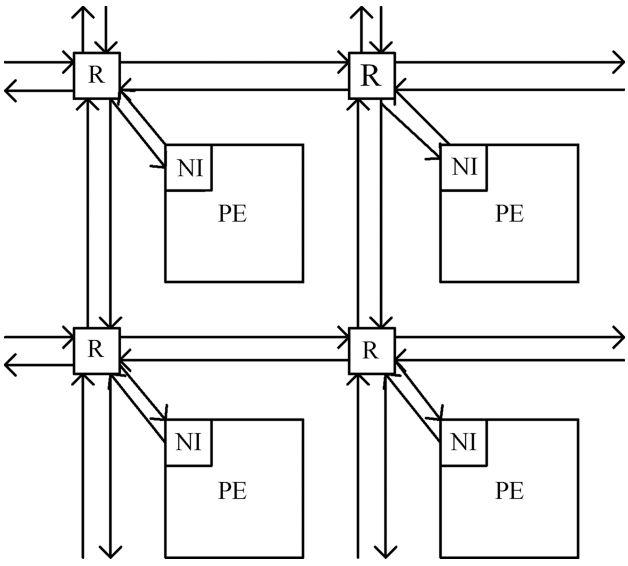


Fig. 1. Structure diagram of the 2D mesh network-on-chip.

* Project supported by the National Natural Science Foundation of China (Nos. 60676009, 60725415, 60971066, 60803038) and the National High-Tech Program of China (Nos. 2009AA01Z258, 2009AA01Z260).
† Corresponding author. Email: guanxuguang_5@126.com
Received 21 October 2009, revised manuscript received 23 December 2009

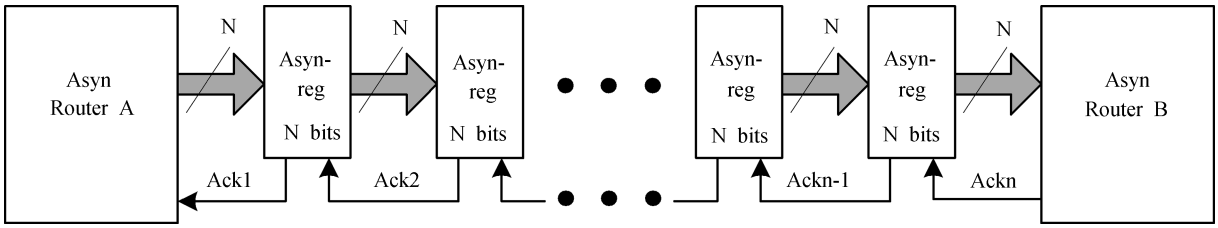


Fig. 2. Conventional transmission channels between routers.

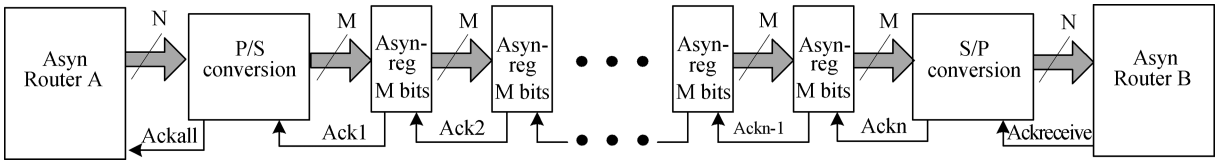


Fig. 3. Proposed byte-level serial transmission channels between routers.

Table 1. Dual-rail quasi-delay insensitive encoding.

D	D0	D1
Data 0	1	0
Data 1	0	1
Null	0	0
Invalid	1	1

2. Conventional transmission manners in network-on-chips

The working manner of network-on-chip refers to the data transmission mode in computer networks. Data can be transmitted to the corresponding target module by route switching, which substitutes the conventional data transmission mode in bus based architecture. So it has a high concurrent transmission capacity and expansibility[9]. Because transmission data in asynchronous on chip networks are reached by handshake signals other than the clock, problems caused by the clock can be eliminated, and modularity is also greatly enhanced[10]. Although on chip networks have varies of topologies, 2D mesh topology is widely used in network-on-chips, and Figure 1 shows its structure. The NoC mainly consists of computational processing elements (PE), network interfaces (NI), and routers. The latter two comprise the primary communication architecture.

Supply noise, electromagnetic coupling and capacitive coupling can all cause variations in the propagation delay of signals, and these effects are often most noticeable in the system-interconnect of large network-on-chips. Asynchronous logic and QDI (quasi-delay insensitive) circuits[11] in particular are attractive for their tolerance of such variations and are gaining acceptance in the NoC space for its tolerance which can translate into power-saving and area advantages.

The most representative example using the quasi-delay insensitive working manner is the CHAIN NoC[12]. It uses dual-rail delay insensitive encoding[13], which guarantees the robustness and transmission efficiency of the circuit. The specific encoding manner is shown in Table 1. Each bit data is represented by two lines. “01” represents logic 0, while “10” represents logic 1. “00” state is a null state, and the state is needed to separate two data cycles. That is, each data transmission must go through a null state before getting into the next transmission cycle. In the dual-rail working manner, data also represent request signals, so control circuit design can be avoided. Furthermore, the dual-rail working manner employs a flow control mode named “back pressure” [14], so its normal working is not susceptible to delay variations. These advantages have made it very popular with asynchronous designers recently[15].

Figure 2 shows the conventional transmission channel between NoC routers. In synchronous systems, it usually uses an even number of inverters to maintain performance if needed over long wire lengths. But in asynchronous systems, it usually uses asynchronous registers to implement this function, as well as some more buffer spaces. It can be found that asynchronous registers in the channel are controlled by handshake signals. Only when the present stage has received the acknowledgement signal from its next stage can it go into the next working cycle. If the next stage is blocked, then the acknowledgement signal from the next stage will be continuously high, and the data is stored at the next stage until the block is removed. So the flow control of the channel is automatically realized. But this transmission method also has its shortcomings, that is, the number of interconnection lines is very large, and these lines not only occupy a very large area, but also consume a great deal of power. Furthermore, a large number of connection lines could make the crosstalk even worse.

So if we can reduce the connection lines between routers, the problems above could be lessened. One of the solutions is to serialize the outgoing data from the router, as shown in Fig. 3. The proposed serial transmission converter is a coarse serial transmission converter, that is, it is not a bit-level serial transmission converter, but a byte-level serial transmission converter. Its advantages are significant. As the parallel data gets more and more serialized, the number of acknowledgement cycles per data increases. One possible way around this is to use a coarser grain acknowledgement that acknowledges at the byte level. So this method not only guarantees the proper performance of the transmission, but also greatly reduces the connection lines between routers. Crosstalk, area and power

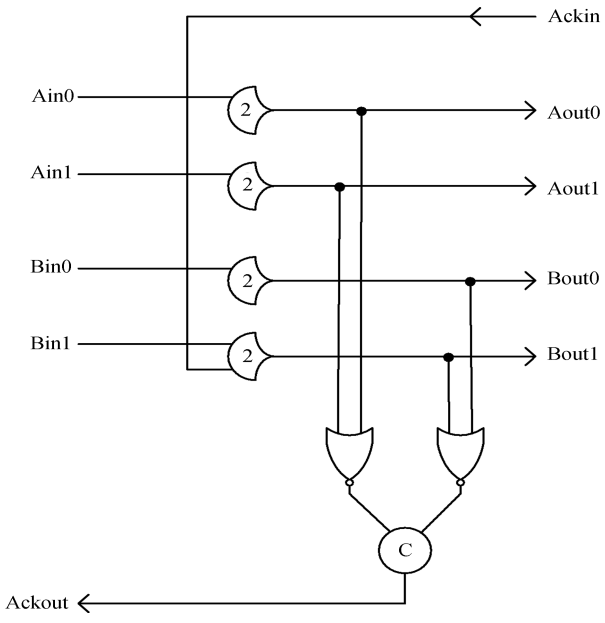


Fig. 4. Conventional dual-rail delay-insensitive asynchronous register (2 bits).

consumption are also reduced, which enhances the stability of the transmission.

3. Proposed serial transmission converter

The design of the asynchronous byte-level serial transmission converter needs to meet two major targets. Firstly, data must be sampled safely, and this is the foundation of correct operation of the circuit. Secondly, at the receiver router, data must be restored from the serial transmission channel intact. We take a 32 bit data flit width as an example in the following circuit diagrams to specify the structure, and null convention logics (NCL) are used here to guarantee the delay-insensitivity of the circuit.

The basic unit in an asynchronous transmission channel is the dual-rail delay-insensitive asynchronous register, as shown in Fig. 4.

The circuit in Fig. 4 has the functions of data storage and completion detection. The assertion of the Ackin signal indicates that the next stage has completed data storage, and asks the present stage for the next data, while the Ackout signal indicates that the present stage has completed data storage, and asks the previous stage for the next data. The Ackout signal is generated by the completion detection circuit of register outputs Aout0, Aout1, Bout0 and Bout1. The acknowledgement signal can not only control the normal working order of the register, but can also implement the flow control through "back pressure". This kind of register can only transmit in one direction, and $2n + 1$ lines are needed when transmitting n bits data.

The parallel data from asynchronous router A have to pass through a parallel/serial transmission converter to become byte-level data. The specific implementation of the parallel/serial transmission converter is shown in Fig. 5.

As can be seen from the diagram, the parallel to serial transmission converter mainly consists of three parts: NCL-register, NCL-MUX and NCL-sequencer4. The first half of the

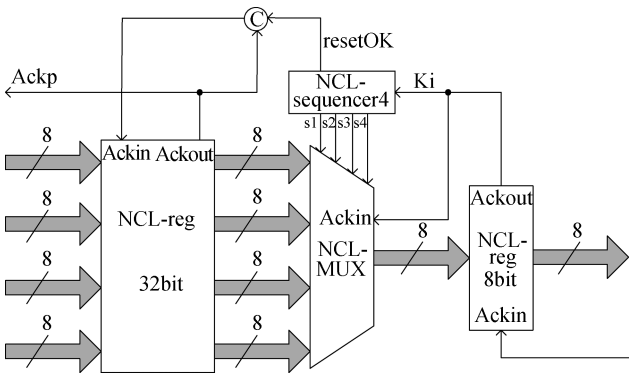


Fig. 5. Proposed byte-level parallel to serial transmission converter.

diagram is the 32 bit NCL-register; it is responsible for transmitting 32 bits data to the NCL-MUX, which is controlled by NCL-sequencer4. The NCL-sequencer4 module can enable the four outputs (s1, s2, s3 and s4) alternately according to the full/empty state of the 8 bit NCL-register in the next stage. The working procedure of the circuit in Fig. 5 is as follows. In the beginning, Router A sends out 32 bits data, and the data will first be preserved in the 32 bit NCL-register. If the 8 bit NCL-register in the last half of the diagram is now free, it will send out a Ki signal to the NCL-sequencer4. The NCL-sequencer4 will enable s1, so the first group of 8 bits data in the 32 bit NCL-register can go through NCL-MUX to the 8 bit NCL-register. After the 8 bit NCL-register has successfully sent the data out, the Ki signal will go low, and s1 will be disabled in the following. Attention should be paid here that only the first group of 8 bits data in the 32 bit NCL-register can not go into the NULL period immediately, and this NULL period will go through NCL-MUX to reach the 8 bit NCL-register. After the 8 bit NCL-register in the last half of the diagram has successfully sent out the NULL period, the Ki signal will go high again, and this time the s2 signal is set to high. Thus the second group of 8 bits data in the 32 bit NCL-register can go through NCL-MUX to the 8 bit NCL-register. The subsequent working procedure is just the same as that above. When the 32 bits data have all passed the NCL-MUX, resetOK and the Ackout signal of the 32 bit NCL-register will go high, and this will cause the Ackin signal of the 32 bit NCL-register to go high. That is, the 32 bit NCL-register has sent out the 32 bits data successfully, and is ready to go into the next working cycle.

When data are transmitted to the target router, a serial to parallel transmission converter is needed to convert the byte-level serial data to parallel data. The specific implementation of the byte-level serial/parallel transmission converter is shown in Fig. 6.

As can be seen from the diagram, the byte-level serial to parallel transmission converter mainly consists of four parts: NCL-register, NCL-DEMUX, NCL-sequencer4 and Completion detection module. The first half part of the diagram is the 8 bit NCL-register; it is responsible for receiving data from the byte-level serial transmission channel. NCL-DEMUX is used to restore the parallel data from serial data, and is controlled by the NCL-sequencer4 module. The NCL-sequencer4 can enable the four outputs (ss1, ss2, ss3 and ss4) alternately according to the request signal of the incoming 8 bit register. Only when

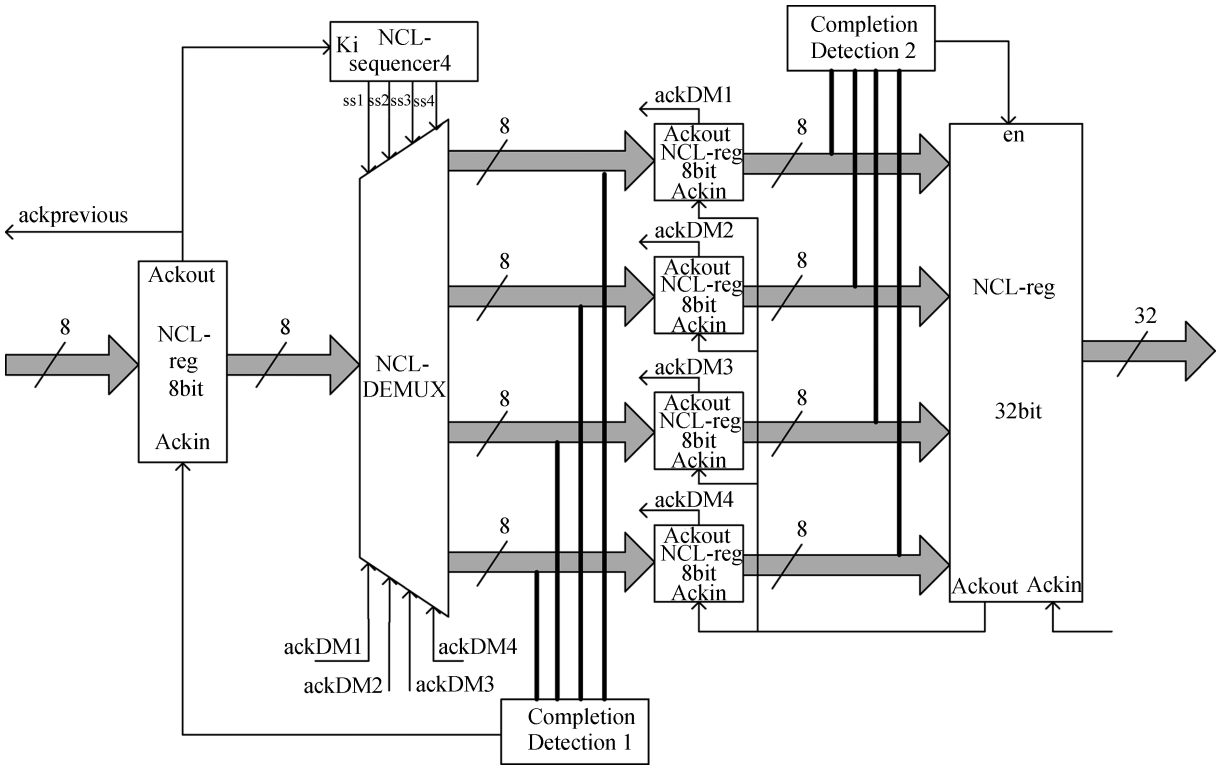


Fig. 6. Proposed byte-level serial to parallel transmission converter.

the four NCL-registers in the middle part of the diagram are filled with data can 32 bits data be transmitted into the next stage, so the data can be restored intact. The specific working procedure of the circuits in Fig. 6 can be depicted as follows. In the beginning, the 8 bit NCL-register sends out a Ki signal to the sequencer4 after it has received the serial data from the transmission channel. The sequencer4 module enables ss1 to control the NCL-DEMUX to put the serial data onto the first 8 bit NCL-register. At the same time, the Completion Detection 1 module will send back an ack signal into the incoming NCL-register, and the incoming NCL-register begins to enter the NULL period. Also, the first 8 bit NCL-register in the middle part of the diagram sends back an acknowledgement signal named "ackDM1" into the NCL-DEMUX. NCL-DEMUX will receive the NULL signal from the incoming 8 bit NCL-register once it has received the ackDM1 signal. The subsequent working cycle is just the same as that above. Attention should be paid here that the incoming data can not go through the last 32 bit NCL-register until the four 8 bit NCL-registers in the middle part are all filled with data, and this is controlled by the Completion Detection 2 module. Unlike the Completion Detection 1 module, the Completion Detection 2 module only asserts when the four inputs are all high. Under these circumstances, the 32 bit NCL-register can receive the data intact.

As can be seen, the sequencer4 module plays an important role in the proper working of the circuit. It controls the working rhythm of the converter. The specific implementation of sequencer4 is shown in Fig. 7.

The sequencer selects the output through the input times of Ki. The last input can be preserved in the circuit due to the hysteresis of threshold gates until input Ki changes, which makes S1, S2, S3 and S4 become high alternately. The specific working process of the sequencer is depicted in Table 2. When Ki goes high for the first time, S1 goes high, while S1 goes low after Ki becomes low. When Ki becomes high for the second time, S2 goes high, and so on. The mutual exclusion behavior of S1–S4 guarantees that each piece of data can only enter the corresponding register, which makes the serial transmission converter work properly.

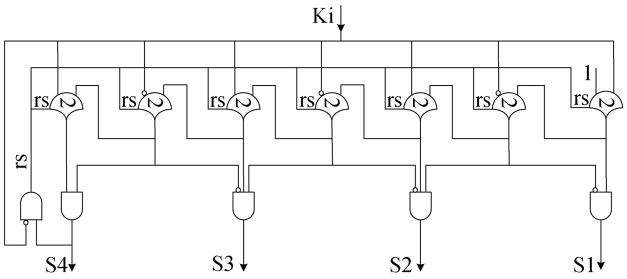


Fig. 7. Proposed sequencer with four outputs.

Table 2. Operating process of sequencer4.

Period	Ki	S1	S2	S3	S4
#1	1	1	0	0	0
#2	0	0	0	0	0
#3	1	0	1	0	0
#4	0	0	0	0	0
#5	1	0	0	1	0
#6	0	0	0	0	0
#7	1	0	0	0	1
#8	0	0	0	0	0

The NCL-DEMUX used in Fig. 8 is not the same as con-

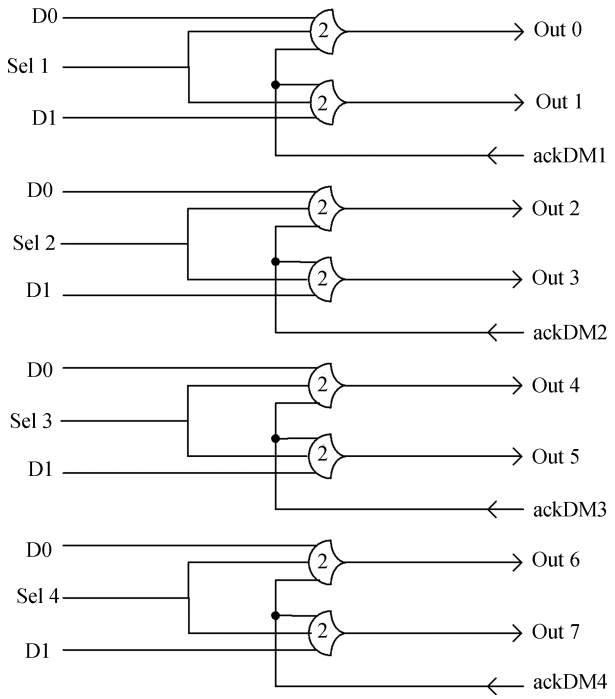


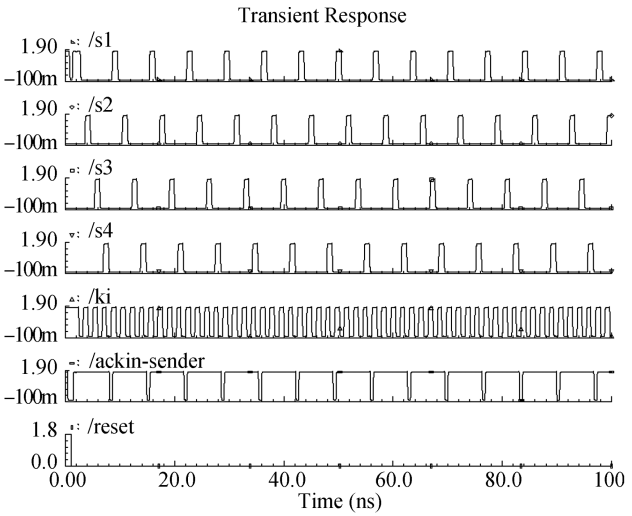
Fig. 8. NCL-DEMUX modules in a byte-level serial to parallel converter (1 bit).

ventional ones. It has several acknowledgement signals; each acknowledgement signal is applied to each input group. Here we take a 1 bit NCL-DEMUX module as an example. If Sel 1 is high, then data D0, D1 can go through threshold gate and get to Out 0 and Out 1. The next stage register will send back an ackDM1 signal if data are successfully received. The merits of separating the acknowledgement signals are significant. The change of acknowledgement signal in the present group does not have an impact on other input groups. Also, the conversion speed of the separated acknowledgement signals is much faster than that of centralized acknowledgement signals.

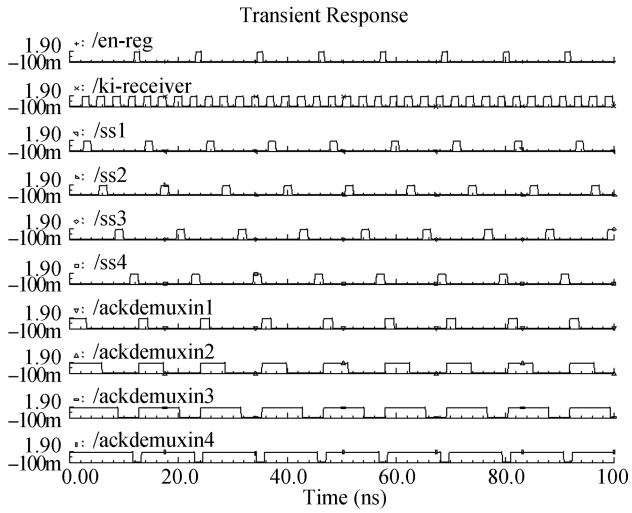
4. Simulation results and analysis

The entire circuits of the serial transmission converter and the byte-level serial channel are implemented using SMIC 0.18 μm CMOS standard technology. Simulations on parallel to serial conversion and serial to parallel conversion were made respectively. Figure 9 is part of the SPICE simulation waveform.

As can be seen form Fig. 9(a), each time ki increases, one of the selection signals (s1, s2, s3, s4) of NCL-MUX goes high, and the parallel data are divided into four parts by NCL-MUX. The ki signal is the acknowledgement signal of the byte-level NCL-register, and its state indicates the full/empty degree of the channel. Only after four parts of the parallel data have passed NCL-MUX can the ackin-sender signal go low. This guarantees that parallel data are not flushed by the next period data before the present data are transmitted into the byte-level transmission channel. In the serial to parallel conversion of Fig. 9(b), the ki-receiver is the request signal extracted from the last byte-level NCL-register, and high in the ki-receiver indicates that the serial data have arrived at the input port of NCL-DEMUX. In the meantime, it notifies the NCL-sequencer4 at the receiver to begin to work. The NCL-sequencer4 asserts four



(a) Waveform of the parallel to serial conversion



(b) Waveform of the serial to parallel conversion

Fig. 9. Simulation waveforms of P/S and S/P conversion.

Table 3. Conversion delay under different technology corners (27 °C, 1.8 V).

	Delay P→S (ps)	Delay S→P (ps)
ss	376.14	691.26
tt	303	569.2
ff	247.08	470.7

outputs (ss1, ss2, ss3, ss4) alternately according to the incoming times of the ki-receiver. Ackdemuxin1 to ackdemuxin4 are four acknowledgement signals from the four NCL-registers in Fig. 6, and ackdemuxin1 to ackdemuxin4 match ackDM1 to ackDM4 in Fig. 6. When the ackdemuxin signal goes low, it indicates that the NCL-register has successfully received the data from NCL-DEMUX. So the ackdemuxin signals go low one by one with the ki-receiver signal, and DEMUX can receive data from the channel intact, as seen in Fig. 9(b). The en-reg signal is the enable signal of the 32-b NCL-register in the latter half of Fig. 6. It guarantees that only when the four NCL-registers in the middle part of Fig. 6 are all filled with data can 32 bits data be transmitted into the next stage. We find that the en-reg signal rises only after the ackdemuxin4 signal goes low, so the

Table 4. Comparison of frequency and throughput under different data flit widths.

	16 bits		32 bits		64 bits	
	Frequency (MHz)	Throughput (MB/s)	Frequency (MHz)	Throughput (MB/s)	Frequency (MHz)	Throughput (MB/s)
Serial	476.19	476.19	446.43	446.43	362.3	362.3
Parallel	378.78	757.6	357.14	1428	334.4	2675.6
Percentage	—	62.9%	—	31.2%	—	13.5%

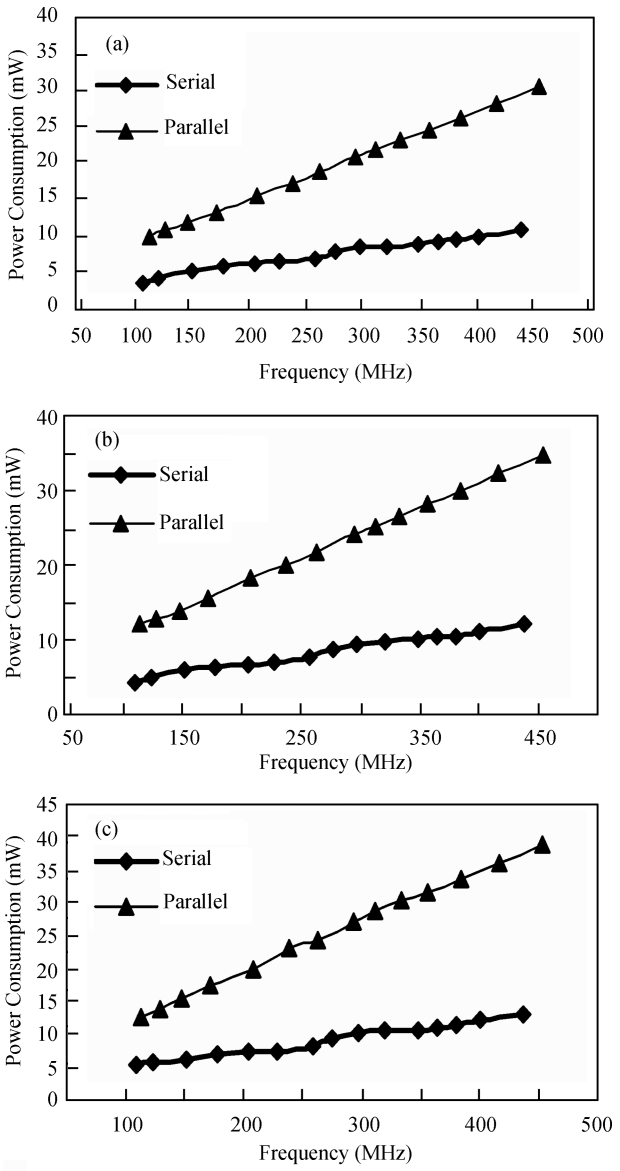


Fig. 10. Power consumption versus working frequency under different buffer stages (32 bits). (a) 4 stages. (b) 5 stages. (c) 6 stages.

data can be transmitted intact.

To test the conversion performance under different technology conditions and the sensitivity to PVT variations, simulations were made on three technology corners (tt, ss, and ff) at 27 °C. The results are shown in Table 3.

Delay P→S represents the delay from parallel data reaching NCL-MUX to byte-level data going into the transmission channel, while Delay S→P represents the delay from serial data converting to parallel data. It can be seen that the circuit can work properly with preferable performance under different variations of technology corners. Varying the technology conditions has little impact on the circuit, namely the circuit has a better robustness.

Table 5. Percentage of power saved under different bit widths and stages at 300 MHz.

	16 bits	32 bits	64 bits
4 stages	31.68%	57.01%	69.87%
5 stages	34.67%	61.1%	72.82%
6 stages	37.16%	62.96%	74.58%

To compare the performance of the proposed byte-level serial transmission converter under different data flit widths, the maximum working frequency and throughput of different data flit widths are given in Table 4. Percentage represents the throughput ratio of the serial transmission manner to the parallel transmission manner.

It can be found that as the data width grows, the working frequencies of the two transmission methods decrease slightly. This is because the loads of acknowledgement signals of each stage increase gradually with increasing data widths, and this point lowers the speed of backward transmission. Due to the average working cases and clockless nature of asynchronous circuits, the asynchronous serial transmission channel can not work as quasi-synchronous ones do, which can speed up in serial transmission by increasing the clock frequency. So this point causes the throughput to decrease with increasing data flit widths. But this also avoids the extra circuits needed in clock speed acceleration in quasi-synchronous network-on-chips. Moreover, low power full asynchronous network-on-chips are mainly focused on power consumption instead of speed, so the throughput of serialized data still can satisfy most of the requirements of low power network-on-chips.

Comparisons of power consumption versus working frequency were made under different buffer stages. Here we test three kinds of data flit widths, 16 bits, 32 bits and 64 bits. The results are shown in Fig. 10 to Fig. 12.

We can find that with increasing working frequency, the power consumption of the two channels increases simultaneously. But the parallel channel increases much faster than the serial channel; this is because the parallel channel has more transistors and interconnection lines, and these two issues make the interconnect overheads much bigger. Furthermore, with the number of buffer stages increasing, we find that the power consumption difference between two channels becomes larger and larger. It can be found that the two lines are closer in the 16 bit data width situation, while the two lines are much further apart in the 64 bit data width situation. So the power saved in the 64 bit data width is much more significant than in the 16 bit data width. Specific results are shown in Table 5. In addition, the

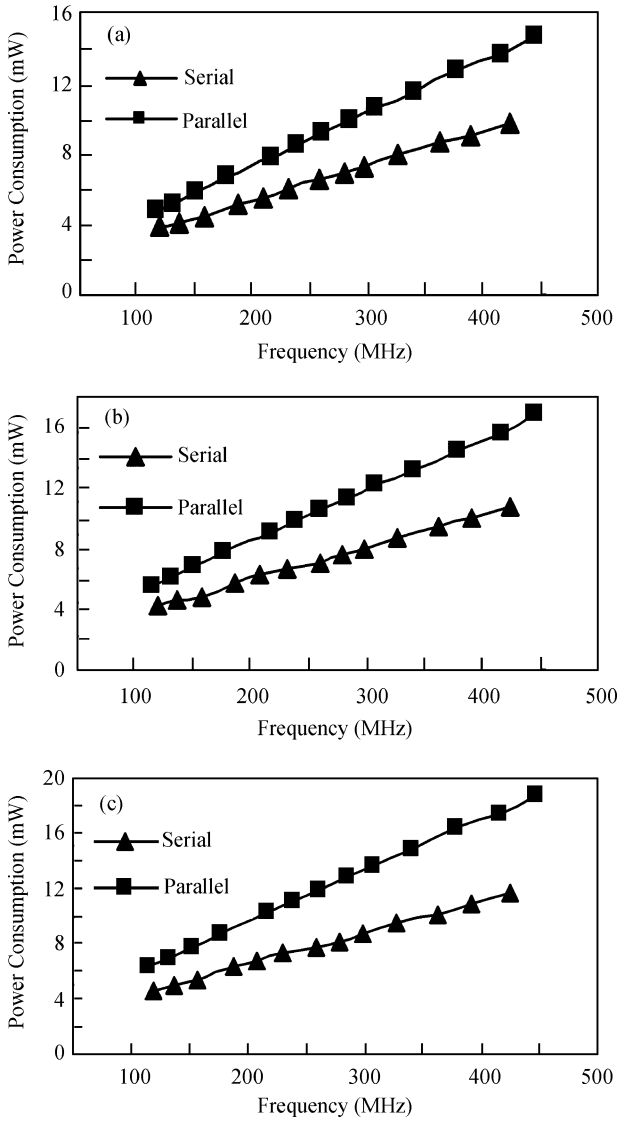


Fig. 11. Power consumption versus working frequency under different buffer stages (16 bits). (a) 4 stages. (b) 5 stages. (c) 6 stages.

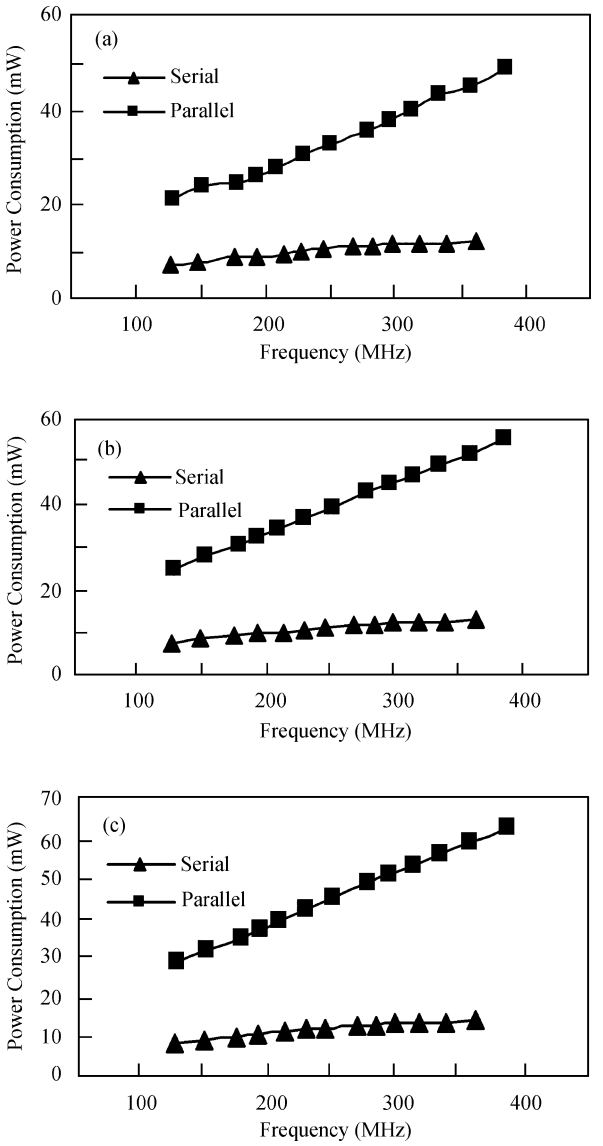


Fig. 12. Power consumption versus working frequency under different buffer stages (64 bits). (a) 4 stages. (b) 5 stages. (c) 6 stages.

more buffer stages in a long interconnection line there are, the more obvious the merits of serial transmission mode are. Generally, more than four buffer stages are often used between on chip routers[16], so the serial channel is superior to the parallel channel in area and power overheads.

The area overheads of the parallel and proposed serial links under different data flit widths are given in Fig.13 and are expressed by gate numbers.

It can be found that the gaps between serial and parallel transmission modes increase with increasing data flit width. Also, we find that with increasing buffer stage, the gates consumed by the parallel channel grow faster than those of the serial channel. In the circumstances of 6 buffer stages and 64 bit data width, the gates consumed by the serial transmission channel are nearly one third of the gates consumed by the parallel transmission channel. So the proposed serial transmission channel has great advantages in area overheads, and the smaller area can also help reduce the power overheads.

Figure 14 shows the percentages of long connection lines saved by two transmission modes under different data flit widths. We find that serialization on 16 bits data can save long connection line resources by approximately 50%, while serialization on 64 bits data can save up to 86% of the long connection lines. Reducing the number of connection lines can directly save on-chip area and can decrease the crosstalk effect to some extent. So the serial transmission channel can show lots of benefits in area and power consumption and lower the complexity in connection line routing.

So the advantages of serialization on different data flit widths are quite disparate. Firstly, serialization on 16 bits data can reduce the throughput loss to a minimum, and the channel can have a relatively higher working frequency. But it is not optimal in power consumption and connection line resources. On the other hand, serialization on 64 bits data can save the most power and connection line resources, but the loss in throughput is much bigger. Furthermore, the time needed for data decomposition and reconstitution is much larger (because the sender needs to decompose 8 times to complete the serialization, and

Table 6. Comparison between the conventional serial converter and the proposed one.

	Working manner	Delay-insensitive	Max working freq @ 32 bit input data flit	Percentage of power saved @ 6 stages, 300 MHz
Proposed	Dual-rail	Yes	446.43 MHz @ 180 nm	62.96% @ 180 nm
Method in Ref. [8]	Single-rail	No	300 MHz @ 120 nm	53.1% @ 120 nm

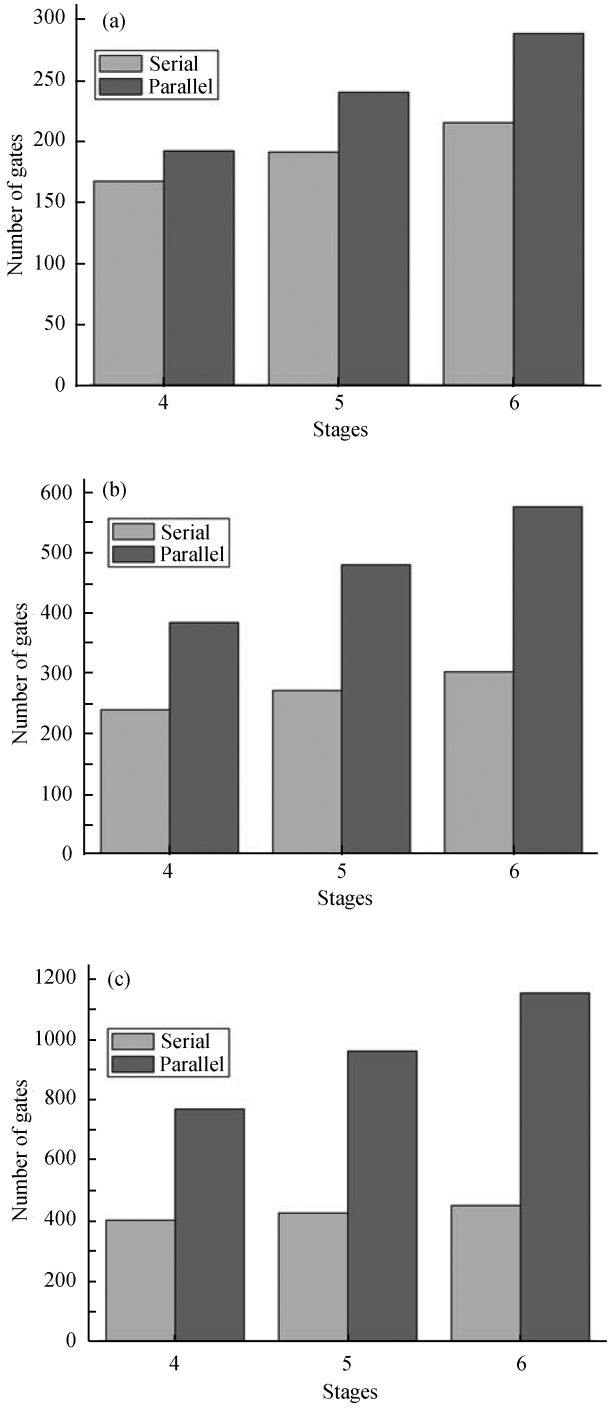


Fig. 13. Comparisons of number of gates consumed by two transmission modes under different bit widths. (a) 16 bits. (b) 32 bits. (c) 64 bits.

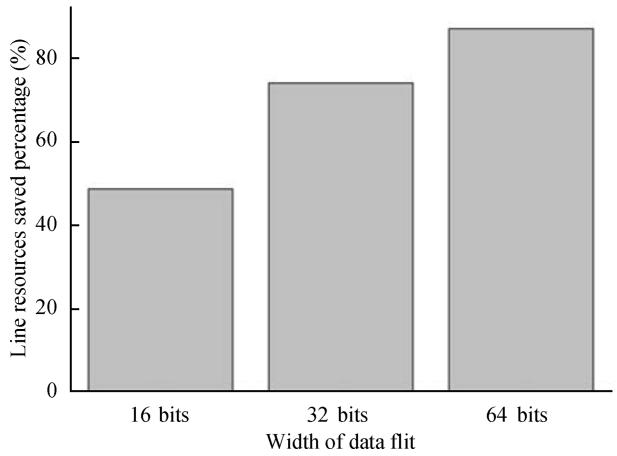


Fig. 14. Comparisons of percentages of line resources saved under different data flit widths using the proposed method.

the receiver needs to reconstitute 8 times to built the parallel data). This indicates that serialization on 64 bits data is not suitable for transmitting real-time signals, and it is only suitable for transmitting data which have lower demands on transmission delay. Serialization on 64 bits data is more applicable to those routing areas which are sensitive to area and connection lines, while serialization on 16 bits data is more applicable to real-time signal transmission due to its lower throughput loss and shorter decomposition and reconstitution time.

Here we compare the proposed serial converter with the conventional one; specific results are shown in Table 6.

The two serial converters are compared in terms of working manner, delay sensitivity, working frequency and percentage of power saved. We find that the proposed one is obviously superior to the conventional one at working frequency and percentage of power saved. Also, the proposed converter does not have to use the synchronous/asynchronous module which is used in Ref. [8], so the power consumption can be further reduced. The dual-rail working manner also enhances the robustness of the circuits and makes the circuit delay-insensitive, while the conventional one uses a single-rail working manner and is vulnerable to physical process variations.

5. Conclusions

This paper proposes a method to improve the shortcomings of large interconnection lines in conventional network-on-chips. By serializing the parallel data into small group of data blocks, the connection lines and area overheads are greatly decreased. Also, the power consumption of the transmission channel is only about one third of the conventional one under 32 bit data flit width. Threshold gates make the circuit quasi-delay insensitive, thus the robustness of the circuit is increased. The proposed serial transmission converter is suitable for full asynchronous network-on-chips which are sensitive to area and power. It is hoped that the proposed method can make a valu-

able contribution to the area of low-power NoCs.

References

[1] Dally W J, Towles B. Route packets, not wires: on-chip interconnection networks. The 38th ACM Design Automation Conf, 2001: 684

[2] Benini L, Micheli G D. Networks on chips: a new SoC paradigm. Computer, 2002, 35(1): 70

[3] Guerrier P, Greiner A. A generic architecture for on-chip packet-switched interconnections. DATE, 2000: 250

[4] Kumar S, Jantsch A, Millberg M. A network on chip architecture and design methodology. IEEE Computer Society Annual Symposium on VLSI, 2002: 105

[5] Bainbridge J, Furber S B. Chain: a delay-insensitive chip area interconnect. IEEE Micro, 2002, 22(5): 16

[6] Lines A. Asynchronous interconnect for synchronous SoC design. IEEE Micro, 2004, 24(1): 32

[7] Morgenshtein A, Cidon I, Kolodny A, et al. Comparative analysis of serial vs parallel links in NoC. International Symposium on System-on-Chip, 2004: 185

[8] Ogg S, Valli E, Al-Hashimi B, et al. Serialized asynchronous links for NoC. Design, Automation and Test in Europe, 2008: 1003

[9] Dobkin R, Ginosar R, Kolodny A. QNoC asynchronous router. Integration, the VLSI Journal, 2009, 42(2): 103

[10] You J, Xu Y, Han H, et al. Performance evaluation of elastic GALS interfaces and network fabric. Electronic Notes in Theoretical Computer Science, 2008, 200(1): 17

[11] Verhoeff T. Delay-insensitive codes, an overview. Distributed Computing, 1998, 3(1): 1

[12] Bainbridge W J, Furber S B. Chain: a delay insensitive chip area interconnect. IEEE Micro Special Issue on Design and Test of System on Chip, 2002, 22(5): 16

[13] Sankar R, Kadiyala V, Bonam R, et al. Implementation of static and semi-static versions of a 24+8×8 quad-rail NULL convention multiply and accumulate unit. Region 5 Technical Conference, 2007: 53

[14] DeMara R F, Kejriwal A, Seeber J R. Feedback techniques for dual-rail self-timed circuits. Proceedings of International Conference on VLSI, 2004: 458

[15] Birtwistle G, Stevens K S. The family of 4-phase latch protocols. 14th IEEE Symposium on Asynchronous Circuits and Systems, 2008: 71

[16] Bainbridge W J, Salisbury S J. Glitch sensitivity and defense of quasi delay-insensitive network-on-chip links. 15th IEEE Symposium on Asynchronous Circuits and Systems, 2009: 35