NIUGAP: Low Latency Network Interface Architecture with Gray Code for Networks-on-Chip

Daewook Kim, Manho Kim and Gerald E. Sobelman
Department of Electrical and Computer Engineering
University of Minnesota, Minneapolis, MN 55455 USA
Email: {daewook,mhkim,sobelman}@ece.umn.edu

***Abstract*— The implementation of a high-performance network-on-chip (NoC) requires an efficient design for the network interface unit (NIU) that connects the switched network to the IP cores. In this paper, we present a novel NIU architecture that utilizes a Gray code based packet reordering methodology to achieve low latency packet processing. The proposed architecture has been implemented with VHDL and synthesized using a 0.25 μm ASIC technology. Simulation results verify the functionality of the architecture and show that it can save a substantial amount of packet processing time compared to the conventional reordering scheme.**

I. Introduction

The increasing complexity of systems-on-chip (SoC) together with the wiring problems of advanced IC technologies make networks-on-chip (NoCs) a promising replacement for buses and dedicated interconnect [1]. The major components of an NoC are switches, which transport data from one place to another, and network interface units (NIU), which implement the interface between IP cores and switches, as shown in Figure 1. The main functions of an NIU are data packetization (packet assembly), depacketization (packet disassembly), end-to-end flow control with scheduling for buffer overflow and protocol coherence between interconnected modules.

Some previous papers have discussed how to implement a packet reordering mechanism in other networking areas, such as in ATM parallel switching [2]–[4]. There are two de facto packet reordering standards widely adopted in current network designs. The first uses sequence numbers [2] and the second uses a time stamp [3]. In [2], an implementation is described using sequence numbers in an ATM switch, along with an assessment of its performance under simulated traffic. Henrion et al in [3] presents a time stamp scheme to address the problems, which requires constant time to process each packet. However, all of the above mechanisms require packets to be saved first and reordered in the switch output buffers before departing to the next destination and the packet reordering methodologies depend on the time sequence represented by a binary-coded decimal (BCD) code. Several comparison steps are necessary between packets to obtain the correct packet order, which causes a serious processing time overhead. That methodology can be appropriate for computer networks like LAN and WAN to link hundreds of switches in a large-scale network. However, those mechanisms are not appropriate for the on-chip switched network platform since the switch overhead in terms of area and packet processing latency have to be minimized as much as possible. Therefore, packet reordering methodologies must be reconsidered for application to an NoC platform. Some researchers have described the functions and importance of their NIUs for macro-networks such as LANs and WANs [5], [6]. Other works [7], [8] discuss NoC-specific NIUs but have not yet sufficiently addressed the issue of lower latency data transactions.

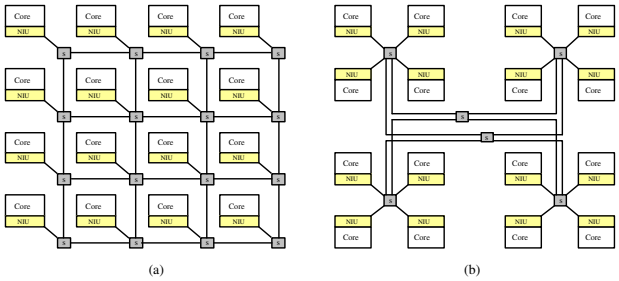

Fig. 1. NoC topologies with NIU. (a) 2D mesh. (b) Butterfly fat tree.

In this paper, we propose a novel network interface architecture called **NIUGAP**, **N**etwork **I**nterface **U**nit with **G**r**A**y code **P**acket reordering. Our model includes a packet reordering function in the NIU which minimizes the switch packet reordering overhead. In particular, the NIUGAP utilizes a proposed Gray code based novel packet reordering scheme to minimize the packet reordering overhead.

The proposed architecture has been implemented with VHDL and synthesis was performed using the Synplify ASIC 3.3 tool with the Chip Express CX4000 structured ASIC library for 0.25 μm technology. Simulation results verify the functionality of the architecture and show that the proposed Gray code based NIU architecture can save significant packet processing time compared to the conventional packet reordering schemes.

II. Packet Reordering with Gray Code

A Gray code is an ordered binary code in which two successive values differ in exactly one bit position. It was originally proposed as a way to prevent spurious outputs from electromechanical switches [9] but has seen widespread application in many applications. In the transition between two states as shown in Figure 3, all three switches change state. In the brief period while all are changing, the switches will read some spurious position and the transition might look like

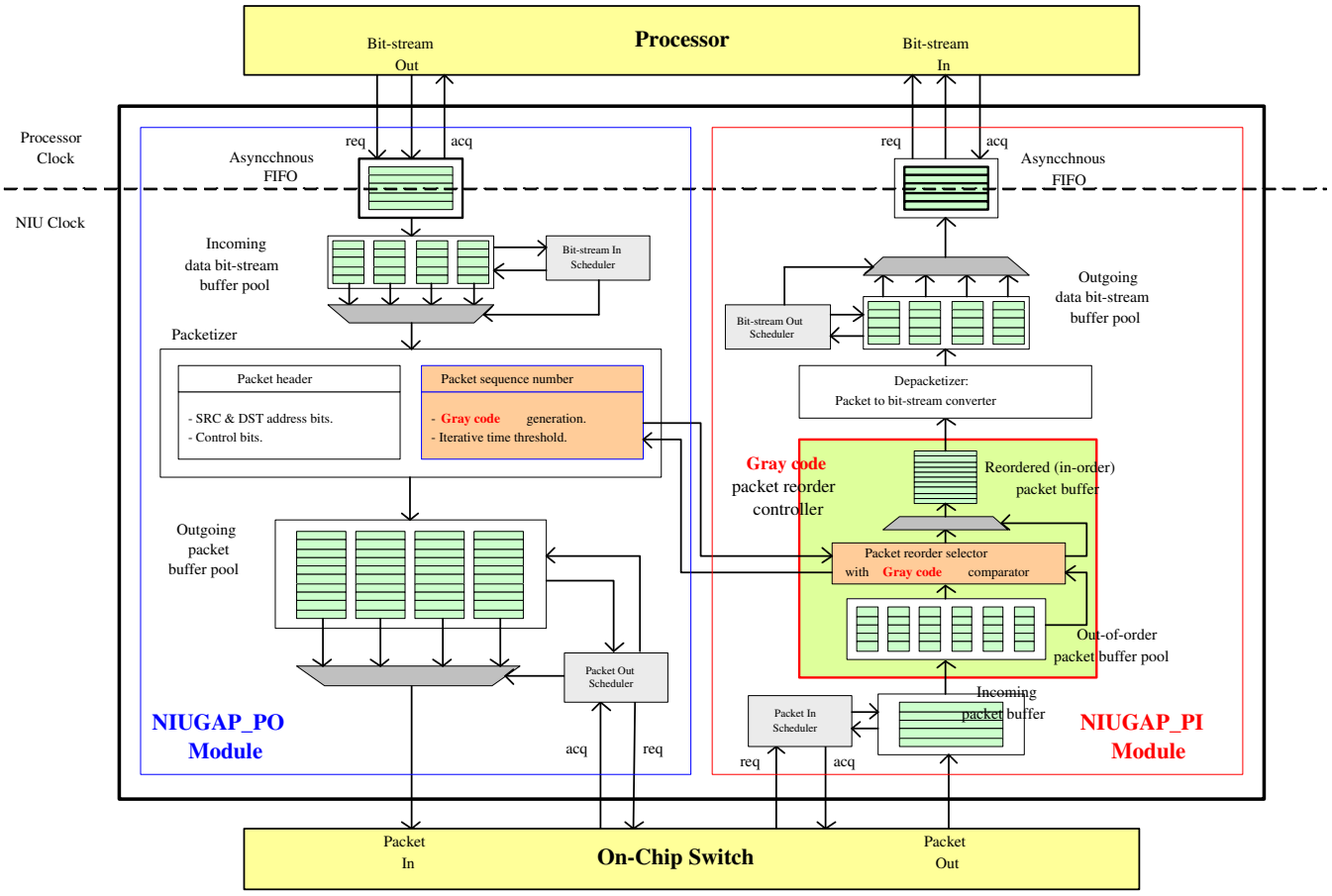

Fig. 2. NIUGAP architecture block diagram.

011-001-101-100. When the switches appear to be in state 001, the observer cannot tell if that is the intended state or merely a transitional state between two other states. A Gray code eliminates this problem by changing only one bit at a time, so there is never any ambiguity about the intended state.

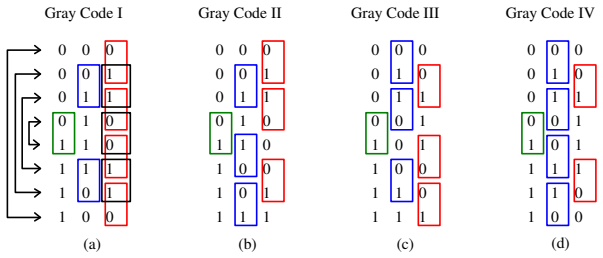

Fig. 3. Various Gray code sets.

As shown in Figure 3, the last state can roll over to the first state with only one switch change. This is called the ”cyclic” property of a Gray code. Several sets of Gray code can be generated having the feature of single bit change. We adopted the pattern of Figure 3(a) because it follows a regular change pattern in its transitions. The most common Gray code, as shown in Figure 3(a), is a reflected code where a bit in any column except the MSB is symmetric about the sequence mid-point. This means that the second half of the Gray code is a mirror image of the first half. A good way to visualize the coding is by noting that the least significant bit follows a repetitive pattern of 2: 11, 00, 11 etc. The second digit follows a pattern of 4. Therefore, we can utilize the regular pattern of Gray code transition to check for an in-order packet arrival sequence quickly by monitoring single bit changes with an XOR check and a repetitive pattern of the reflected Gray code set, which prevents other disordered Gray codes with the same Hamming distance of 1 from being regarded as the expected correct code. Figure 4 shows the 4-bit BCD code for 16 decimal digits and the logic for converting from BCD code to Gray code.

III. NIUGAP Architecture

NIUGAP is a network interface architecture specifically designed for NoCs which achieves fast packet reordering through the use of a Gray code. Figure 2 shows the overall block diagram of the proposed NIUGAP architecture.

A. NIUGAP_PO Module

As shown in the NIUGAP Packet Out (NIUGAP_PO) module in Figure 2, all bit-stream data that is generated from the processor is entered into the NIU by synchronizing the

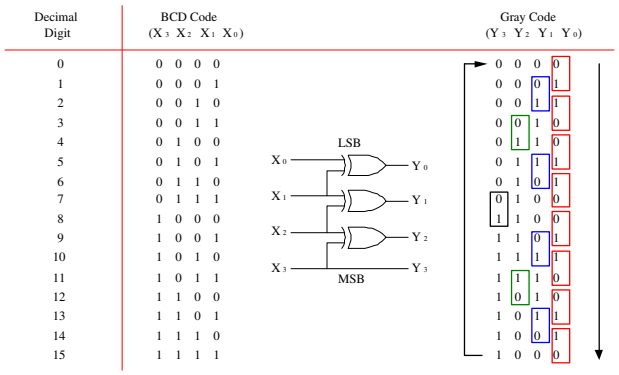

Fig. 4. BCD code to Gray code conversion.

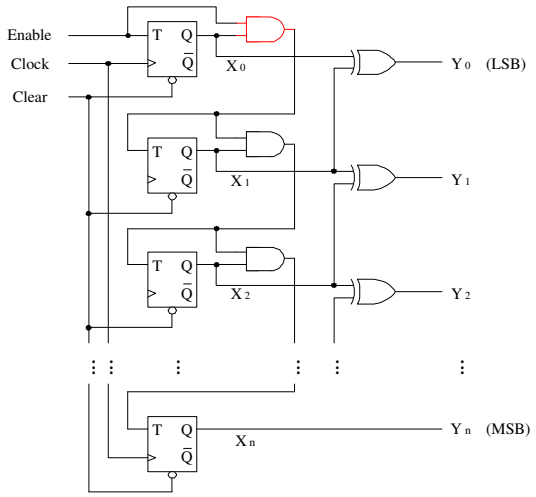

Fig. 5. Gray code counter block diagram.

processor clock with the NIU clock through an asynchronous FIFO. The synchronized bit-stream data is saved in the incoming data buffer pool, and then every fixed number of bits are assembled into the packet format. Packet header information such as source address, destination address and control bits are added to the actual payload for packet routing. Furthermore, an m-bit time tag and an n-bit sequence tag represented by the Gray code are also added to the header information for the purpose of packet reordering at the destinations. We can reuse every n-bit reflected Gray code recursively by changing the m-bit time tag offset, which is used for the iterative time offset for packet transmission. All generated packets are temporarily saved in the outgoing packet buffer pool, and then each scheduled packet is transmitted to the on-chip switch by maintaining a simple req/ack protocol. Figure 5 shows the logic block diagram for the n-bit recursive Gray code counter we propose. The Gray code counter changes its contents on every clock pulse. Often it is desirable to be able to inhibit counting, so that the count remains in its present state. This may be accomplished by including an Enable control signal. Connecting the Enable signal to the first AND gate chain means that if Enable=0, then all T inputs will be equal to 0. If Enable=1, then the counter operates normally. In many cases, it is necessary to start with the count equal to zero. This is easily achieved if the flip-flops can be cleared.

B. NIUGAP_PI Module

The NIUGAP Packet In (NIUGAP_PI) module in Figure 2 shows the detailed data flow of incoming packets out of the on-chip switch to the processor in a bit-stream format. Incoming packets from the on-chip switch are saved in the incoming packet buffer temporarily, and then the packets are moved to the Gray code packet reorder controller block. The proposed packet reordering based on a Gray code is performed in that block. Reordered packets are sent to the depacketizer for packet to bit-stream conversion. All disassembled bit-streams are saved in the outgoing bit-stream buffer pool, and then those are transmitted to the processor by synchronizing the clock through an asynchronous FIFO with a req/ack protocol. Figure 6 describes the functional operations of the Gray code packet reorder controller. The m-bit time tag field of the packet header indicates any time threshold offset given when a set of packets are generated from the source during some time interval. The time tag field is also represented by a Gray code. The sequence tag field is for the sequential Gray code generated during the corresponding time tag. An iterative n-bit Gray code is generated and added onto the packet header every new time tag. Therefore, we can also reorder the out-of-order packets that have the same sequence tag by checking the time tag information, although this would be a very unusual case. The packet headers that include a Gray code based time tag and sequence tag are compared by the Gray code comparator checking for a single bit change and a regular transition pattern of a Gray code set as described above. When the expected packets or retransmission packets arrive late, those can take the bypass path for preferential packet transition to avoid performance degradation.

IV. Experimental Results and Performance Analysis

We present the implementation results for an RTL-based NIUGAP architecture. We have used the Chip Express CX4000 structured ASIC library for 0.25μm technology, and synthesized with the Synplify ASIC 3.3 tool. The architecture overhead is determined in terms of gate count and area. The design was successfully simulated and verified with the ModelSim simulator using the post-synthesis netlist and randomly generated packet patterns. Table I provides values for the total packet transmission latency for the 88-bit packet size (3-bit Gray coded time tag, 12-bit Gray coded sequence tag, 3-bit BCD coded SRC, 3-bit BCD coded DST, 3-bit BCD coded control bit, and 64-bit BCD coded payload fields). We present the optimal estimated frequency for the packet size that avoids negative slack from the synthesis netlist.

Figure 7 and Figure 8 presents the average packet latency comparison for the Gray code based packet reordering vs. the typical BCD code based packet reordering methodology by varying the size of time tag plus sequence tag fields of the

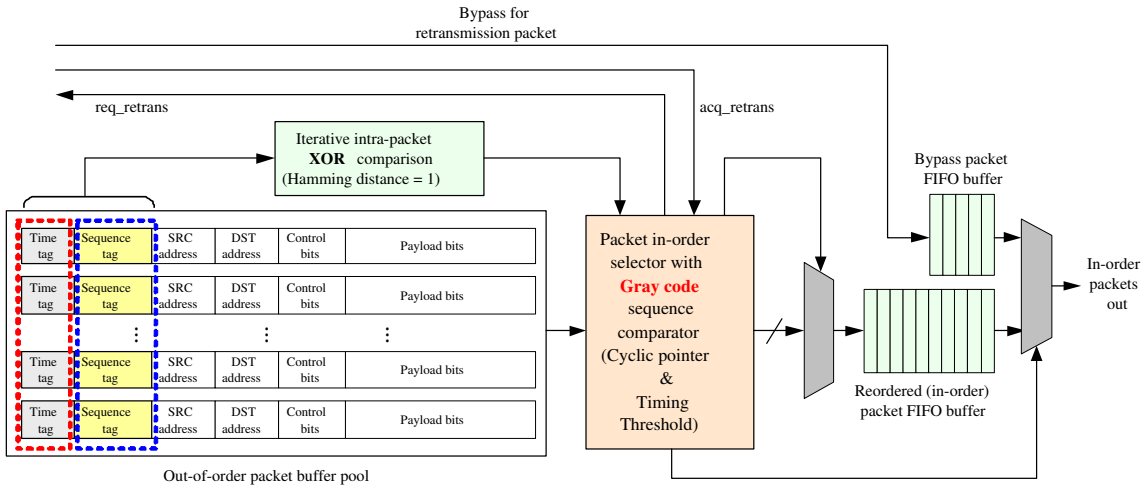

Fig. 6. Gray code packet reordering controller block diagram.

TABLE I
SYNTHESIS AREA AND TIMING REPORT

Components	*0.25μm ASIC Technology*		
	Gate count	*Area Overhead*	*Optimal Estimated Frequency*
NIUGAP_PO Module	2644	2718.6 μm^2	46.8 MHz
NIUGAP_PI Module	5342	16583.4 μm^2	46.8 MHz

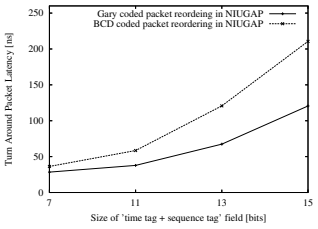

Fig. 7. Packet latency for NIU.

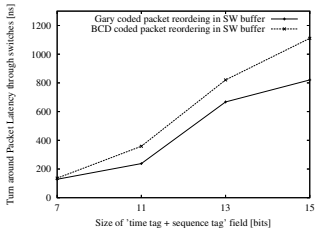

Fig. 8. Packet latency through switches.

packet. Figure 7 shows the latency overhead performed inside NIUGAP block. Figure 8 presents the corresponding latency overhead that is executed by adapting the two mechanisms into the switch output buffer as used in macro-networks. The latency was calculated for the overall delay time through two switches. The simulation results show the proposed Gray coded packet reordering methodology outperforms in both cases the conventional BCD coded methodology.

V. CONCLUSIONS

We have presented a novel on-chip NIU architecture that utilizes a Gray code based packet reordering methodology for the purpose of low latency packet processing. The proposed architecture has been implemented with VHDL and synthesized using a 0.25 μm ASIC technology. The simulation results verify the function of the architecture and show that the proposed Gray code based NIU architecture can save significant packet processing time compared to the conventional packet reordering scheme.

ACKNOWLEDGMENTS

We thank Sangwoo Rhim, Bumhak Lee, and Euiseok Kim of the SAMSUNG Advanced Institute of Technology (SAIT) for their help with this manuscript. This research work is supported by a grant from SAIT.

REFERENCES

[1] J. Henkel, W. Wolf, and S. Chakradhar, "On-chip networks: a scalable, communication-centric embedded system design paradigm," in *IEEE Proc. of 17th International Conference on VLSI Design*, Jan 2004, pp. 845–851.

[2] J. Turner, "Resequencing cells in an ATM switch," *Washington University, Computer Science Department, WUCS-91-21*.

[3] M. Henrion, "Resequencing system for a switching node," *U.S. Patent #5,127,000*.

[4] I. Keslassy and N. McKeown, "Maintaining packet order in two-stage switches," in *INFOCOM 2002. Twenty-First Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings*, Feb 2002, pp. 1032–1041.

[5] P. Sundstrom and P. Andersson, "ATM network interface architectures for low latency," in *IEEE Computer Communications and Networks, 1997. Proceedings., Sixth International Conference on*, Jan 1997, pp. 494–499.

[6] R. Osborne, Q. Zheng, J. Howard, R. Casley, D. Hahn, and T. Nakabayashi, "Dart - a low overhead ATM network interface chip," in *Proc. Hot Interconnects*, Sep 1996, pp. 587–593.

[7] A. Radulescu, J. Dielissen, S. Pestana, O. Gangwal, E. Rijpkema, P. Wielage, and K. Goossens, "An efficient on-chip NI offering guaranteed services, shared-memory abstraction, and flexible network configuration," vol. 24, Jan 2005, pp. 4–17.

[8] P. Pande, C. Grecu, A. Ivanov, R. Saleh, and G. D. Micheli, "Design, synthesis, and test of networks on chips," in *IEEE Proc. of Design and Test of Computer*, March 2005, pp. 404–413.

[9] F. Gray, "Pulse code communication," *U.S. Patent #2,632,058*.