CONGESTION CONTROL ALGORITHM FOR NETWORK ON CHIP TRAFFIC

Vazgen Melikyan, Aram Martirosyan, Arshavir Matevosyan, Samvel Chobanyan

SYNOPSYS ARMENIA CJSC, Tigran Mets Avenue 49, 375010 Yerevan, Armenia, phone: +374 1559583, e-mail:vazgenm@synopsys.com

The problem of providing congestion free traffic flow control in Network on Chip (NoC) , which found wide application nowadays in System on Chip (SoC) design area is examined. Congestion control algorithm for NoC traffic is proposed and performance of algorithm is evaluated through cycle-accurate simulation of NoC example built on SystemC components.

Keywords: System on Chip, Network on Chip, traffic flow control

1. INTRODUCTION

Design of Systems on Chip is complex, multidimensional task, which is becoming more challenging with technology scaling. The number of processing units integrated on single chip has increased substantially in past decade and will continue to increase with further technology scaling. According to International Technology Roadmap for Semiconductors number of transistors on SoCs will grow up to 4 billion transistors operating at 10GHz frequency till the end of current decade [1]. Providing reliable communication between processing units of SoCs with current methods [2] will become extremely complex and in some cases impossible. Proceeding from this consideration in the beginning of current decade Network on Chip a new SoC design paradigm have been proposed [1]. NoCs are layered, packet-switched interconnection networks integrated onto a single chip and their operation is based on operating principle of macro-networks. Several NoC design methods and synthesis tools were developed recently [3,4]. Mesh topology NoC is shown in figure1.

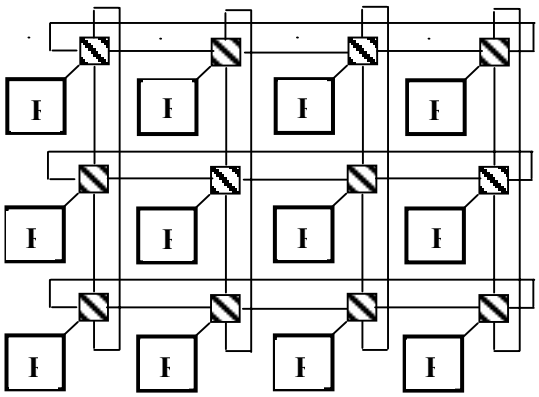

Figure 1 Mesh topology NoC

Blocks named R in figure 1 represent resources. Resource consists of IP block and network interface which provides communication interface between IP block and network switch. Lined blocks in figure 1 are switches, which are building components of communication fabrics between resources.

2. NoC Flow Control Overview

In general the purpose of flow control algorithm is in regulation of data packets traffic, in order to avoid congestion and resource starvation in the network. Two flow control approaches are widely adopted in NoCs - switch-to-switch and end-to-end flow control algorithms. Figure 2 illustrates congestion control mechanisms which are implemented in above mentioned algorithms.

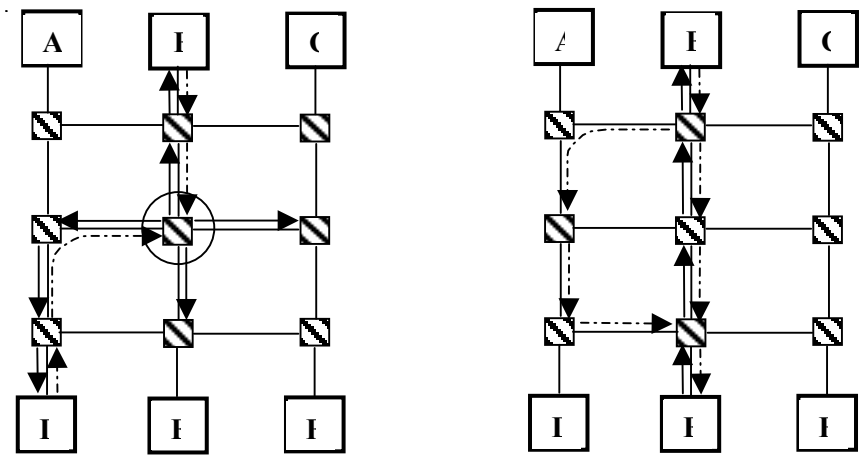

Figure 2 Switch-to-switch (a) and end-to-end (b) flow control mechanisms

In figure 2 blocks from A to F are resources and lined blocks are switches. Dotted arrows represent data traffic and solid arrows represent feedback information which is sent by switches to their neighbors informing about switch state. Suppose that in figure 2a resources B and D transmit packets which must pass through middle switch (switch in the circle). Congestion occurs when buffering resources of switch are full and switch can not receive incoming packets until buffers are full. In switch-to-switch flow control mechanism switches periodically inform about their state to neighboring switches, which make routing decision according to state of neighboring switches. State of switch is defined by concrete realization of network, like stress values defined in [5], or another parameter which gives information about switch loading. In figure 2a when congestion occurs middle switch send data about it's state to neighboring switches, those in turn send data to their neighbors and before congestion information reaches to traffic source it continues data transmission, which can bring to more severe congestion. This is the main drawback of congestion control mechanism implemented in switch-to-switch flow control algorithm.

In figure 2b end-to-end congestion control mechanism operates as follows. Traffic injection rate is regulated directly at the source of message (resource B in figure 2b). Traffic source can transmit new packet of data only after feedback (solid arrows in figure 2b) packet from receiver resource (resource E in figure 2b) will inform it about receiving of current packet. That means that packets are transmitted in turn, new packet enters to network only after old one leaves it. Negative aspect of this congestion control mechanism is sufficient overhead when sending feedback packet.

Taking into consideration drawbacks of above described congestion control mechanisms another algorithm of congestion control is developed. One of advantages of proposed congestion control algorithm is that it regulates traffic directly in the network during data transmission. Algorithm stops data transmission according to loading information collected from switches. Every switch makes decision about rerouting packets or stopping data transmission individually unlike switch-to-switch congestion control mechanism described above. This gives opportunity to stop data transmission in switch before congestion information will reach to resource and not to continue loading buffers of switch which is under risk of being congested. Second advantage is that it doesn't suffer from large overhead like end-to-end flow control algorithm.

3. PROPOSED FLOW CONTROL ALGORITHM

In proposed flow control algorithm every switch contains counters. Counters keep the number of empty input buffers for every direction in the switch. In figure 3 C_L, C_R, C_T, C_B are counters correspondingly for left, right, top and bottom directions. In every clock cycle switch is sending data stored in counters to neighboring switches and in this way all switches have information about state of neighboring switches, which gives opportunity to reroute packets and even stop packet injection. Architecture of switch is shown in figure 3.

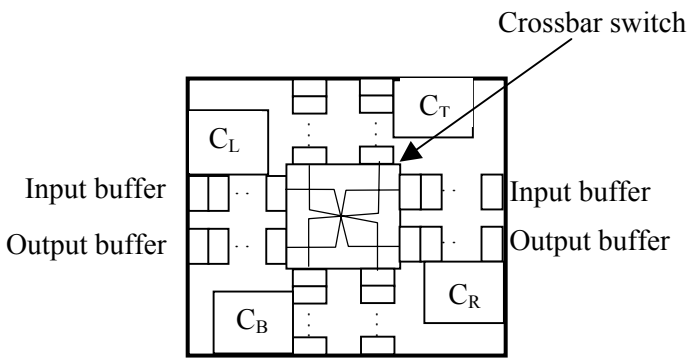

Figure 3 Architecture of NoC switch in proposed method

Examination of algorithm is carried out using example shown in figure 4. In figure 4 dotted arrows illustrate traffic paths from resources A_2 to C_2, from A_2 to D_2, from C_3 to B_2 and correspondingly from B_2 to C_3. Solid arrows illustrate data path from resource B_1 to C_2 which must pass through switch in circle. Since path to C_2 resource is busy by traffic injected from A_2 (dotted arrows) algorithm must reroute packet from B_1 or must halt traffic injection from B_1 to avoid congestion in switch in the circle. Algorithm operates as follows. Starting point for our examination is the state of network when resource A_2 is transmitting packets to C_2 and D_2 and communication between B_2 and C_3 already exists (dotted lines in figure 4). As was mentioned above all switches have information about state of neighboring switches and resources have the state information of switches that are connected directly to

them. Resource B_1 starts packet transmission after checking buffer availability of switch S_1, if it has at least one free buffer then B_1 starts packet transmission.

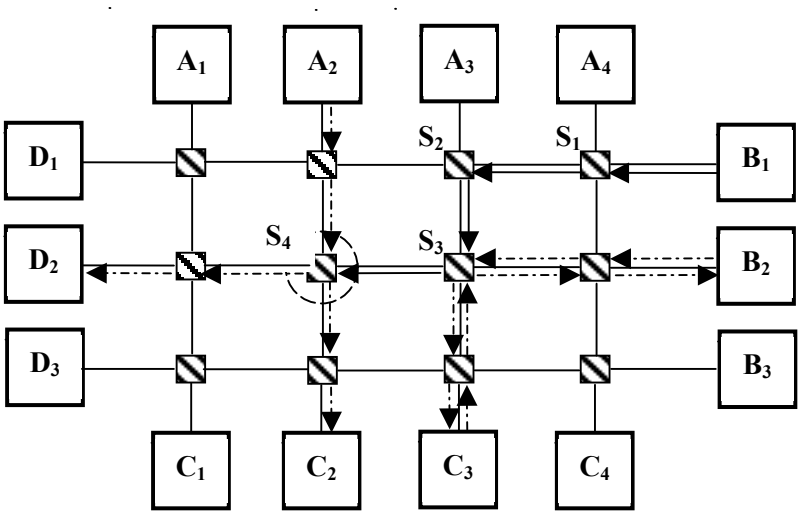

Figure 4 Proposed method of congestion control (case study)

After decoding first received flit of packet (wormhole traffic flow is considered), which contains destination address of packet, S_1 routes packet to S_2 switch. S_2 in it's turn routes packet to S_3 switch. As it is shown in figure 4 the only direction S_3 can route packet is switch S_4. The output of S_4 that corresponds to input from S_3 provides path for traffic flow from A_2 to D_2 and so packets from S_3 can not be transferred until transmission between A_2 and D_2 will not be completed, therefore packets coming from S_3 will be stored in input buffers of S_4. Information about loading of input buffers is sent from S_4 to S_3, from S_3 to S_2 and so on till B_1 resource in every clock cycle. In presented approach it is considered that unit of buffer size is equal to flit size and clock cycle is equal to time period when a flit passes one switch. These considerations gives opportunity to stop traffic injection to switch buffers when only one buffer left empty. When state of right input buffer (number of empty buffers) of S_4 is equal to 1 S_3 stops data transmission and send HALT packet to S_2, S_2 to S_1 and S_1 to resource B_1. After receiving HALT packet every switch stops data transmission, hence escaping congestion before HALT packet will reach resource B_1. In this manner packet transmission from B_1 halts until loading of right input buffer of S_4 will decrease. The following is parameters definition and algorithm description for above described case study.

// Definition of parameters

a) F_{B1} is number of free buffers in right pin, which is stored in network interface of B_1 resource
b) $F_{S1}=[F_L, F_R, F_T, F_B]$ is array of numbers of free buffers of left, right, top and bottom neighboring switches, stored in switches

c) C_L, C_R, C_T, and C_B are number of free buffers for corresponding inputs in the switch which are stored in counters.

1. // Start of data transmission from B_1 resource
   if F_{B1}>1 then
   send first flit of the packet to S_1 switch;
   else
   halt data transmission
2. // Data transmission through S_1 switch
   // After reading first flit packet address is decoded and routing decision is made
   // (in described case packets are transmitted to S2 switch)
   if F_{S2}[R]>1 then
   send data to S_2
   else
   send data to any other direction with more then one empty buffer
   else
   send HALT to B_1 and stop data transmission
3. // Data transmission through S_2 switch
   repeat steps of point 2 for switch S_2
4. // Data transmission through S_3 switch
   repeat steps of point 2 for switch S_3
5. // Storing buffers in right input buffers of S_4
   // Storing incoming flits from S_3 until C_R>1 (C_R corresponds to right input pin
   // of S_4)
   if C_R=1 then
   send HALT to S_3
6. After receiving HALT from S_4, S_3 stops data transmission and send HALT to S_2.
7. After receiving HALT from S_3, S_2 stops data transmission and send HALT to S_1.
8. After receiving HALT from S_2, S_1 stops data transmission and send HALT to B_1.
9. B_1 stops data transmission after receiving HALT from S_1

4. SYSTEMC SIMULATION RESULTS AND CONCLUSION

Cycle-accurate simulation of NoC, that was built using SystemC components was performed. Traffic generators were used to model application traffic. Simulated NoC includes SystemC models of 16 IP blocks and 15 switches to organize communication between them. According to simulation results proposed congestion control mechanism provides congestion free traffic flow in the network at the cost of average latency increase by about 20% compared to NoC operation without using algorithm. Analysis of cycle-accurate simulation results showed that proposed method of congestion control in NoCs can be used in designs with high traffic

injection rates to avoid congestion in network at the cost of acceptable increase in average latency.

5. REFERENCES

[1] Benini L., G. De Micheli, *Networks on Chips: A New SoC Paradigm* IEEE Computer, January 2002, Volume 35, Issue 1, Pages 70-78.

[2] Benini L., D. Bertozzi, *Network-on-chip architectures and design methods* IEE Proceedings Computers and Digital Techniques, March 2005, Volume 152, Issue 2, Pages 261-272.

[3] Jalabert A., S. Murali, L. Benini, G. De Micheli, *xpipesCompiler: A tool for instantiating application specific Networks on Chip* Proceedings of the DATE'04 Conference, 2004, Pages 1-6.

[4] Murali S., G. De Micheli, *SUNMAP: A Tool for Automatic Topology Selection and Generation for NoCs* Proceedings of DAC 2004 Conference, Pages 914-919.

[5] Nilsson E., M. Millberg, J. Oberg, A. Jantsch *Load distribution with the Proximity Congestion Awareness in a Network on Chip* Proceedings of the DATE'03 Conference, 2003, Pages 1126-1127.