C-018

Hardware Design and Cost Evaluation of a Speculative Selection Router

TRAN CONG SO, SHIGERU OYANAGI AND KATSUHIRO YAMAZAKI

Graduate School of Science and Engineering, Ritsumeikan University.

1. Introduction

We have proposed the Speculative Selection Routing (SSR) that improves the capability of selection function for various routing functions [1]. An SSR router would utilize idle cycles of the network links to relay its own condition to adjacent routers (normally the router utilization values), thus, the condition exchange process does not add noticeable delay to the network operation.

The SSR router has given messages high selection flexibilities to dynamically adapt with the changing network condition, thus, requiring quite complex hardware to accomplish. In this study, we evaluate the hardware design of an SSR router for 2D torus network. The router is designed based on *-Channels routing function [2], and then the SSR selection is applied. This study describes the speculation hardware to give a more clear view of the hardware implementation. Moreover, we try to estimate the required hardware cost for implementing SSR into *-Channels in a quantitative approach that gives the number of logic gates required for the selection computation.

2. The SSR Routing algorithm

In this paper, the router is evaluated with the *-Channels routing function in the base with SSR selection algorithm. The *-Channels routing algorithm [2] is a simple and flexible routing function which uses quite small number of virtual channels. The *-Channels algorithm divides virtual channels into two sets of the star channels and non-star channels. Messages move through star channels as when doing the dimension-order oblivious routing. The non-star channels are used when taking any turn that would not be allowed by the dimension-order routing algorithm, thus obtaining full adaptivity without deadlock.

The SSR selection algorithm uses a set of input status to calculate the output channel [1]:

- The link's (virtual channel) availability F= {free, busy}.
- The link's dimension D: $D = 1$ if the message came from the same link's dimension, otherwise $D = 0$.
- The link's current request number R
- The next-node LUCs (Local Utilization Counter) belong to the link values N_1 and N_2, which are corresponding one-hop-far and two-hop-far values.

The selection function evaluates $E = D + R + N_1 + kN_2$. With the two-hop configuration, $k = 0.5$ to decrease the weight of N_2. The best channel is the channel with the minimal E value if there are multiple free channels or all channels are busy.

3. Hardware Design of a SSR Router

3.1 Router Structure

Figure 1 shows the router structure implementing for *-Channels on the 2D torus network. The Input and Output circuitries control virtual channels to adjacent routers. These virtual channels and the network condition buffers share the same physical channel in time-sharing manner. To minimize the latency of the router, multiple routing function (RF) circuitries are placed in Input circuitries to decode messages header and give out the requests to selection function circuitry. Based on local information and network condi-

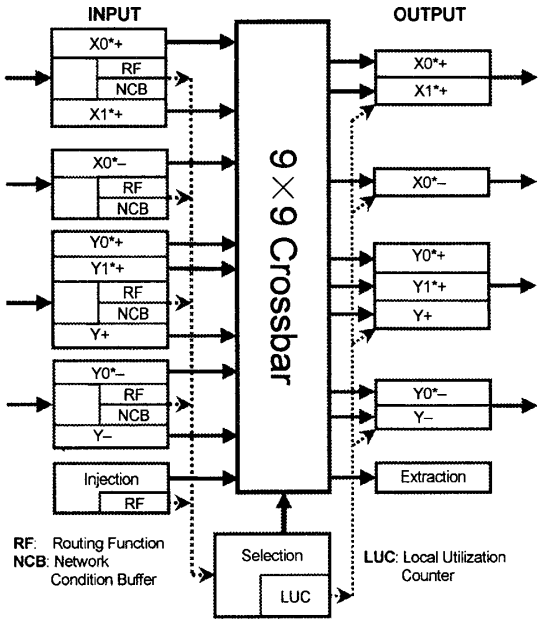

Fig.1 Router Structure

tions, the selection function actually commands the Crossbar to connect an Input to appropriate Output or queues that command until the requiring output is available.

In each router, we have a Local Utilization Counter (LUC) to record how many messages are currently staying in the router. LUC counts up when a message reaches the node and counts down when the message leaves the node. The Network Condition Buffers (NCBs), which is located in Input circuitry, stores other nodes LUCs [1]. In the exchange process, the outputs transfer the value of LUC and 1-hop NCB to corresponding inputs.

3.2 Input Buffers and Network Condition Buffers

This section illustrates the structural design of the Input circuitry on the X+ as an example. Other Input circuitries have the same design with only variation in the number of virtual channels.

Followed the routing function, the virtual channel control signals manage the rotation between virtual channel buffers to receive flit data. On the exchange period, the exchange control signals in the link request the control of link's bus, then direct the two NCBs in the X+ input circuit to receive 1-hop and 2-hop values in sequence. After that, the link's bus is returned to the control of VC control signals. Whenever a message arrives to the current router, the input circuitry provides an signal to LUC in the Selection unit to increase current router LUC value, as to decrease it when the message leave the router. The 1-hop NCB is directly sent to the Selection unit as also to X+ output to be relayed to the next router. The 2-hop NCB is shifted right 1 bit to calculate the kN_2 value and then the value is sent to the Selection unit.

3.3 Selection Unit

The Selection unit is depicted in Fig.2. The SSR router has the same Output Channel Request Queuing Table (Arbitrator) as other oblivious router that controls the requests from the Input circuitry.

The Request Queuing Table is connected to Evaluation Value Computation (EVC) components, which, in turn, calculate the

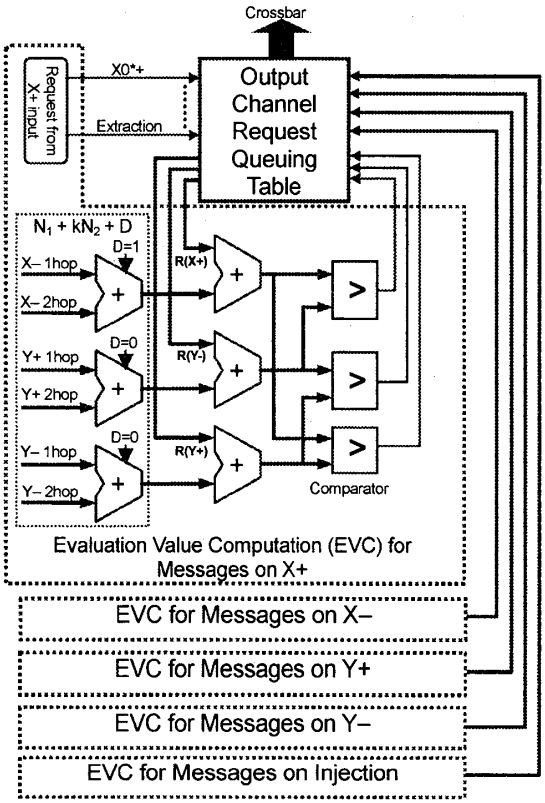

Fig.2: Selection Hardware

evaluation value $E = D + R + N_1 + kN_2$.

The hardware structure of the EVC of the X+ input is shown in Fig.2. Messages coming from X+ input would leave the router in the pair of (Y+, X+) or in the pair of (Y–, X+) because the *-Channels routing function is minimal. So that, the Selection unit needs to compare the E values between Y+ and X+ output or to compare the E values between Y– and X+.

We need to notice that the 1-hop and 2-hop values that are located in X– input are the network statuses for the next nodes on the X+ output direction, as also the Y+ and Y– input stores the network statuses of the next nodes on the Y– and Y+ directions, respectively. The first stage of adders calculates $D + N_1 + kN_2$ and then, in the second stage of adders, the results are added with the number of requests R values, which are calculated by Request Queuing Table, of corresponding outputs. Noticed that D = 1 on the adder that adds the 1-hop and 2-hop values (located in X– input) of X+ output, as the X+ output is in the same dimension that messages come from, and D = 0 with other outputs. Finally the Request Queuing Table takes the comparison results to decide the selection, queues the requests and commands the crossbar connections.

Other EVC computation components in Fig.2 for inputs of Y+, Y– and X– have the same construction while the Injection component is an exception which requires the E values from all other four inputs Y+, Y–, X+ and X– to be calculated.

4. Hardware Cost Evaluation

The two NCB buffers are added to each input block along with several control signals. The fixed shifter for 2-hop NCB value can be implemented very simply by wiring that costs no logic at all. The output blocks of the router only need the virtual channel multiplexer to be modified for an additional channel, and then a multiplexer is used to select the LUC value or corresponding 1-hop NCB value to be transferred to the next adjacent node. Thus, the state machine that controls the information transmission would be simple too.

Table 1: Hardware Cost Estimation

	SSR Computation	*9×9 Crossbar*
2-input LUT	340	–
3-input LUT	20	72
4-input LUT	–	441
XOR	215	–
1-bit Multiplexer	330	–
5-bit Multiplexer	–	144
Flip-Flop	80	–
Total	985	657
No. of Slices	**180**	**273**

The largest added hardware stays in the EVC components in the selection block. With the current unclean straight design, the section requires 32 adders and 16 comparators to do the job. The change to Request Queuing Table is minimal also as it is required to supply four R values to the computational components.

Table 1 gives an estimated hardware cost required for SSR implementation in this router design. Comparative blocks were designed with Verilog HDL and then complied into Xilnix's Spartan II XC2S200 FPGA chip using ISE Foundation tools. We assume that the data width of buffers and adders is 1-byte which would be more than enough since the value of E would never exceed 32 in this case of router (the maximum value depends on the number of virtual channels which decides the maximum number of messages that possibly stay inside the router). For comparative use, the hardware cost for the 9×9 Crossbar in Fig.1 is provided. As Table 1 shows, the logic count of the SSR computation appears higher than the Crossbar, but we do notice that, the Crossbar uses more complex logic units (such as 4-input gate and 5-bit multiplexer) than the hardware for SSR selection. On the other hand, the SSR hardware requires a small number of Flip-flops. In total required area, the SSR requires only 65% the number of slices compared with the Crossbar.

5. Conclusion

This paper presented the hardware implementation method and cost evaluation for Speculative Selection of a 2D torus router. The SSR implementation appears to be simple and straight forward design. The cost estimation shows that the SSR selection hardware would cost less than other important units of the router, such as crossbar unit. In addition, the designing of arithmetic units such as adders and comparators in the speculative selection is not difficult with currently available design methodology of using HDL and EDA tools. Moreover, the SSR hardware can be designed more smartly to reduce hardware cost and to improve the speed of the router.

References

[1] T.C.So, S.Oyanagi, and K.Yamazaki, "Speculative Selection in Adaptive Routing on Interconnection Networks", IPSJ Transactions on Advanced Computing Systems, Vol.44, No. SIG11 (ACS3), pp147-156, 2003.

[2] P.E.Berman, L.Gravano, G.D.Pifarré, J.L.C.Sanz, "Adaptive Deadlock- and Livelock-Free Routing with all Minimal Paths in Torus Networks", Proc. of the fourth annual ACM symposium on Parallel algorithms and architectures, pp.3-12, 1992.

[3] H.Hosogoshi, O.Mitobe, T.Yoshinaga, and M.Sowa, "Design of a Fault-Tolerant Fully Adaptive Router", Symposium on Advanced Computing Systems and Infrastructures (SACSIS2003), IPSJ Symposium Series, Vol.2003, No.8, pp.53-56, 2003