Implementation of Scheduling Algorithms for On Chip Communications

U.Saravanakumar
Assistant Professor/ECE
PSG College of Technology
Coimbatore – 04, India

K.Rajasekar
Assistant Professor/ECE
PSG College of Technology
Coimbatore – 04, India

R.Rangarajan
Principal
Indus College of Engineering
Coimbatore – 101, India

ABSTRACT
Network on Chips (NoCs) replace traditional busses in highly integrated Multiprocessor System on Chips (MPSoCs). As SoCs, communication issues take much important in NoCs but they need to give contention free architecture with low latency. To meet the above need several methods like handshaking mechanism and arbiter designs developed and implemented. This paper presents various scheduler designs using iSLIP scheduling algorithms and its comparative analysis with various arbiters. All the arbiters described using Verilog HDL and synthesized using Xilinx. For performance analysis, Cadence RTL compiler with UMC 0.13µm technology used to compute power and area of all the algorithms for arbiter.

Keywords
SoCs, MPSoCs, Communication latency, scheduling algorithms, arbiter.

1. INTRODUCTION
Ever increasing number of circuits and modules introduce System on Chip which consists of both analog and digital. Shared bus architectures had used to communicate among all the modules in SoC using time division technique caused communication latency. As technology scaled down, processors like Integrated Intellectual Property (*i*IP) blocks have been increased, multiprocessor SoC emerged [1]. On the other hand, designed MPSoCs need to meet shortest communication latency, less energy consumption for global wires and less design time. Additionally, points observed in traditional SoCs are: 1) signal integrity due to different voltage islands and frequencies, crosstalk, electromagnetic interference and soft errors 2) reliability. Based on the above considerations, designers encouraged to investigate and implement network logic on MPSoC caused Network on Chip (NoC). Therefore distinguished features of NoCs are impossible of SoCs. Figure 1 shows the general architecture of NoC [2], [3].

The basic elements of NoC are core processors, core interface, network interface, router and physical links. Topologies for NoCs are mesh, ring, tree, mesh of tree and butterfly, and most preferable topology is mesh. Heart of the NoCs is router and it consists of buffer, crossbar, arbiter and allocators, which coördinate data transfer from input to output based on information received from allocators [4]. Mostly five input and output routers are used for NoC, among five, four I/O ports for adjacent routers and one I/O port for core processor. Buffers are temporarily held the data and release to crossbar which works same as switch. Among all these basic modules, data flow controls of channels whether it may be virtual channels or wormhole play a vital role to give high throughput. NoC routers should provide high speed data transmission when multiple data packets from different input ports to same output. A fast arbiter is most dominant factor in high throughput NoCs. So for the above reason, this study gives analysis of different arbiters on scheduler.

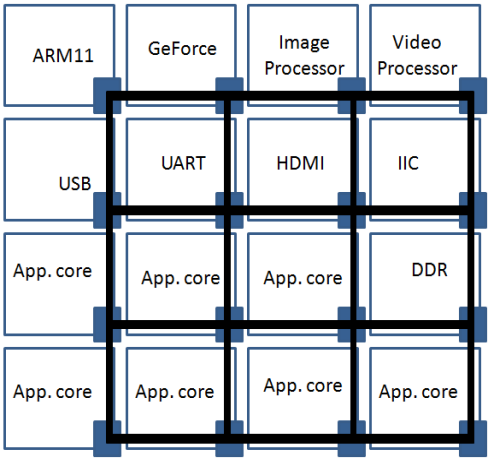

Fig 1: Generic NoC architecture

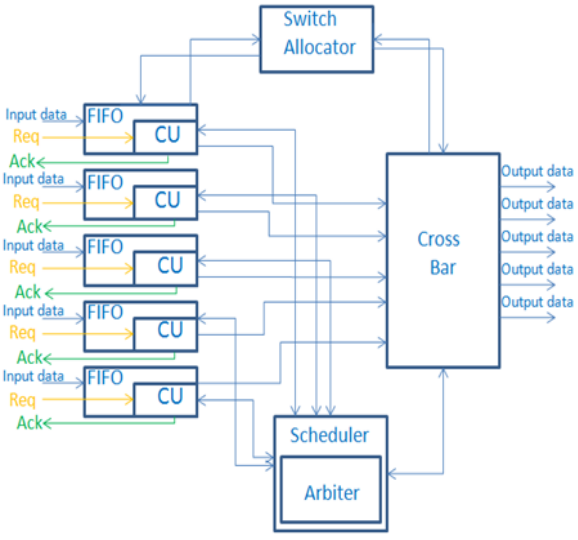

Fig 2: Architecture of NoC router

Figure 2 describes the general architecture of NoC router. This paper includes details of RRA and HRRA in chapter 2, scheduler deals in chapter 3, chapter 4 gives analysis of different arbiter performance on scheduler and chapter 5 provides conclusion of our work [5].

2. ARBITER

The arbiters play important role in designing of schedulers. To design a scheduler, we use two arbiters for grant and accept process. But both the arbiters are identical except mechanism of priority state determination. In this chapter we will see two types of arbiter logic called Round Robin and Hierarchical Round Robin.

2.1 Round Robin Arbiter (RRA)

The main goal of RRA is to produce grant signal for one request among multiple requests from input ports at a time in such a way. A general RRA consist of two blocks, one is input selector and other one is pointer updater. Consider a N input request signal from source port, and possibility to generate grant signal (E) is $log_2(N)$. Pointer updater has a pointer (RPT) that would be generated a grant signal for next possible request from input port in next grant generation signal cycle. Input selector generates a grant signal based on priority, i.e., it gives highest priority to pointer output. An optional signal can be included to indicate no request. Figure 3 shows the block diagram of RRA [6 - 8].

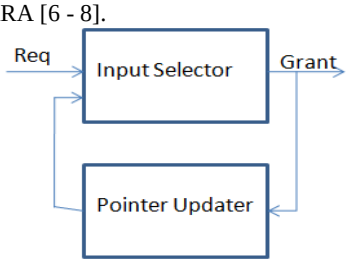

Fig 3: Round Robin Arbiter structure

2.2 Hierarchical RRA (HRRA)

In order to achieve flexibility of priority changing and circular priority order, hierarchical approach is applied in RRA with increased number of request input. In HRRA, first process is to divide number of inputs in to k subsets and which are called as Sub RRA, and they have local request. Local requests and grants are done in multiple stages of sub RRA based on priority settings. A simple pass signal is used for smooth transitions between sub RRAs. Figure 4 describes architecture of HRRA. HRRA also has SPT same as RPT in RRA to point next possible request in queue. HRRA performs arbitration in two stages (Sub RRA_0 and Sub RRA_1).

First Stage (Sub RRA_0): Inputs of every sub RRA is connected to input requests of HRRA, and they select one request in round robin fashion based on priority set by SPT_0. If no request is received by Sub RRA_0, NoReq signal is enabled. One simpler signal Down Request (DRq) is used to differentiate request input signal of Sub RRA_0 and signal of SPT_0. Additionally Select and Pass signals are used to describe selected request by Sub RRA_0s and just granted request by Sub RRA_0s.

Second Stage (Sub RRA_1): DRq signals of every Sub RRA_0 are connected to Sub RRA_1. Selected signals of sub RRA_0s are shown at select output of Sub RRA_1. And Sub RRA_1 has 4-bit granted output to indicate which request signal has granted with corresponding enabled bit. If no request is received by HRRA, the NoReq signal will be low [9].

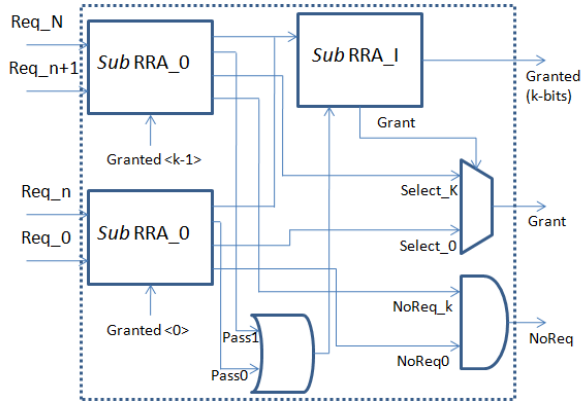

Fig 4: Hierarchical RRA

3. SCHEDULER

Scheduler acts as a central switch arbiter and it analyzes the requests in queue. It configures input ports and interconnects to connect input and output ports, and allows serial data transfer. The scheduler performs large number connections simultaneously based on such algorithms, also avoids conflicts of multiple inputs to single input. Figure 5 shows scheduler implementation chosen for this design. Input to the scheduler is the occupancy vectors from each of the input_blocks with packets waiting to be scheduled. Main part of the scheduler is arbiter. The scheduler consists of two set of arbiters, one to perform grant function, and another to perform accept. The number of grant and accept arbiters depend on number of modules/processors to be fabricated in NoC. Here we designed scheduler using RRA and HRRA for grant and accept process. Scheduler also has an 8-bit busy input from each of the switch outputs. Output port is disabled from the Grant Arbitration if the busy signal is asserted [10].

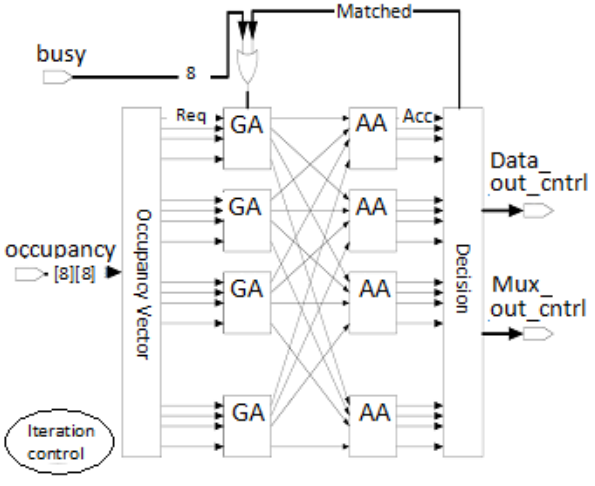

Fig 5: Architecture of Scheduler

In this paper we used *i*SLIP algorithms to function scheduling process, which is developed from SLIP to provide multiple iterations. The SLIP algorithm is a single iteration process, after the iteration is done, possible inputs and output remains unused. Steps of iSLIP algorithm follows as:

Step 1: Request. Each unmatched input sends a request to every output for which it has a queued cell.

Step 2: Grant. In an unmatched output receives any requests, it chooses the one that appears next in a fixed, round-robin schedule starting from the highest priority element. The output notifies each input whether or not its request was granted. The pointer to the highest priority element of the round-robin schedule is incremented to one location beyond the granted input if the grant is accepted in Step 3 of the first iteration.

Step 3: Accept. If an unmatched input receives a grant, it accepts the one that appears next in a fixed, round-robin schedule starting from the highest priority element. The pointer to the highest priority element of the round-robin schedule is incremented to one location beyond the accepted output only if this input was matched in the first iteration.

This scheduler can extend to any type of topology with N number of modules or processors in NoC [11].

4. RESULTS AND DISCUSSION

As discussed above, to provide congestion free and efficient data transfer from multiple inputs to single output, such process plays vital role called scheduling. The scheduler has a main part called arbiter which is used to predict possible request based on priority and gives grant signal. In this paper we designed scheduler with RRA and HRRA. The performance metrics of scheduler are analyzed with RRA and HRRA and they are listed in Table 1, 2 and 3.

Table 1. Power measurement of arbiters and schedulers

Power (nW)	RRA	HRRA	iSLIP Scheduler with RRA	iSLIP Scheduler with HRRA
LP	366.59	157.75	4647	3438.18
IP	53777.35	15507.9	431100.21	322830.74
NP	24919.15	14505.7	250682.53	200269.08
SP	78696.5	30013.6	995685.08	917002.15
TP	157759.59	60184.9	1682114.88	1443539.97

LP Leakage Power
IP Internal Power
NP Net Power
SP Switching Power
TP Total Power

Table 2. Area of arbiters and schedulers

Area (µm2)	RRA	HRRA	iSLIP Scheduler with RRA	iSLIP Scheduler with HRRA
Total Cell Area	1756.32	841.61	22233.09	18017.73

Table 3. Delay measurement of schedulers

Time delays (ns)	iSLIP Scheduler with RRA	iSLIP Scheduler with HRRA
Minimum delay	0.0302	0.03
Maximum delay	0.3921	0.3788
Total delay	2.391	2.001
Slew rate	4.1849	3.1972
Total arrival time	82.624	78.612

First we designed arbiter with round robin fashion and hierarchical round robin fashion, and then scheduler with RRA and HRRA. Scheduler and all the sub modules described using Verilog HDL and functionally verified using Xilinx Isim. Figure 5 shows the simulation output of iSLIP scheduler with HRRA. Other performance parameters were calculated using Cadence RTL compiler with 130nm technology. Technology library file used for calculation is fsc0h_d_sc_bc from Europractice. Major part of the scheduler is arbiter and it consumes more power than other modules. So it insisted us to find and analyze arbiter performance in terms of area, power and delay. Generally scheduler designed with arbiter based on round robin fashion. So for, we also designed RRA and HRRA individually, and measured their parameters like total power of RRA is 157759.59 nW and 60184.9 nW, total cell area is 1756.32µm^2 and 841.61 µm^2 respectively. Therefore by applying hierarchical scheme in arbiter design, 38.15% of power was reduced and 47.92% of area was reduced. Detailed power parameters are listed in Table 1.

In order to get low area, low power and high speed scheduler, we designed iSLIP scheduler with HRRA. To understand the efficiency of proposed design, it is compared with iSLIP scheduler with RRA. From synthesis report observed total power values are 1682114.88nW for iSLIP with RRA and 1443539.97nW for iSLIP with HRRA. Area and other timing parameters are listed in Table 2 and 3.

5. CONCLUSIONS

This paper proposed a scheduler with hierarchical round robin scheme to provide low area, low power and high speed. To achieve the above, hierarchical approach was applied to arbiter design because arbiter is a heart part of the scheduler. Scheduler provides contention free data transfer from input port to output port when multiple requests from input ports to

single output port at a same time. The proposed logic described by Verilog HDL and synthesized using cadence RTL compiler in 130nm technology. With the hierarchical approach to arbiter, area of scheduler is reduced by 18.96% and power of scheduler is reduced by 14.18%.

6. ACKNOWLEDGMENTS

We express our sincere thanks to anonymous reviewers for their valuable suggestions. We also extend our thanks to VLSI Design centre, PSG College of Technology, Coimbatore, India for providing software supports.

7. REFERENCES

[1] International Technology Roadmap for Semiconductors. http://www.itrs.net/

[2] Tobias Bjerregaard and Shankar Mahadevan. "A survey of research and practices of network-on-chip", *ACM Computing Surveys*, 38(1), 2006.

[3] William J. Dally and Brian Towles. Route Packets, Not Wires: On-Chip Interconnection Networks. In *Proc. of the 38th Design Automation Conference (DAC)*, June 2001.

[4] W. J. Dally. Wire-Efficient VLSI Multiprocessor Communication Networks. In Paul Losleben, editor, Proceedings of the Stanford Conference on Advanced Research in VLSI. MIT Press, March 1987.

[5] William J. Dally and Brian Towles. Principles and Practices of Interconnection Networks. Morgan Kaufmann, 2003.

[6] Zhizhou Fu; Xiang Ling, The design and implementation of arbiters for Network-on-chips, Industrial and Information Systems (IIS), 2010 2nd International Conference on , vol.1, no., pp.292-295, 10-11 July 2010.

[7] Yun-Lung Lee; Jer Min Jou; Yen-Yu Chen, A high-speed and decentralized arbiter design for NoC, Computer Systems and Applications, 2009. AICCSA 2009. IEEE/ACS International Conference on , vol., no., pp.350-353, 10-13 May 2009.

[8] Eung S. Shin, Vincent J. Mooney III and George F. Riley, Round-robin Arbiter Design and Generation, ISSS'02, Japan, 2002.

[9] Baranowska, A.; Kabacinski, W, Hierarchical round-robin matching for virtual output queuing switches, Telecommunications, 2005. advanced industrial conference on telecommunications/service assurance with partial and intermittent resources conference/e-learning on telecommunications workshop. aict/sapir/elete 2005. *Proceedings*, vol., no., pp. 196-201, 17-20 July 2005.

[10] Minagar, A.R.; Safavi, S.M, The optimized prioritized iSLIP scheduling algorithm for input-queued switches with ability to support multiple priority levels, Telecommunications, 2003. ICT 2003. 10th International Conference on, vol.2, no., pp. 1680- 1685 vol.2, 23 Feb.-1 March 2003.

[11] N. McKeown, "The islip scheduling algorithm for input-queued switches," Networking, IEEE/ACM Transactions on, vol. 7, no. 2, pp.188–201, Apr 1999.

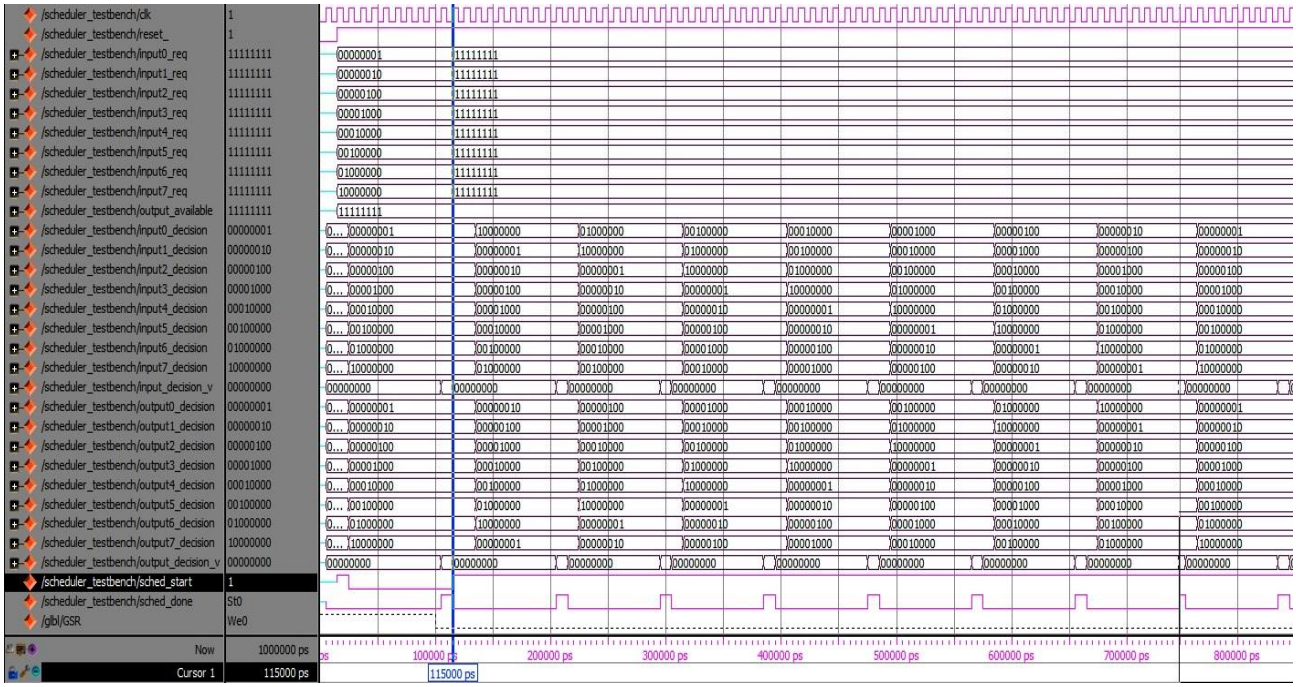

Fig 6: Simulation result of HRRA based iSLIP scheduler