International Journal of Engineering Research in Electronic and Communication Engineering (IJERECE) Vol 3, Issue 4, April 2016

Real Time NoC Based Pipelined Architectonics With Efficient TDM Schema

[1] Laila A, [2] Ajeesh R V
[1] PG Student [VLSI & ES] [2] Assistant professor,
Department of ECE, TKM Institute of Technology, Kollam
[1] lailaslam@gmail.com [2] ajeeshavi@gmail.com

Abstract:-- **As a result of increased demand for on chip communication bandwidth due to multi core chips, packet switched network on chip (NoC) has been emerged as a replacement of bus based on chip interconnects. This work represents a globally asynchronous locally synchronous (GALS) NoC architecture with hard real time multiprocessor platform.It is a special NoC architecture, which uses statically scheduled time division multiplexing (TDM) to control the communication over a structure of routers, links and network interfaces to offer real time guarantees. It also manages the pipelined data maintenance, storage mechanism as well as transfer of data bits from one local memory to another. The router in the architecture uses source routing along with an efficient TDM scheme which does not need any flow control, arbitration and buffering. Also it uses a two phase bundled data handshake latches based on the mousetrap latch controller with a gating mechanism to reduce the energy consumption. The network interface(NI) in the architecture consist of a direct memory access (DMA) controller which avoid the need of synchronization. NI plays an important role in the end to end data transfer between two communicating processor cores. The work is coded by using verilog and is simulated by Xilinx ISE 13.2.**

***Index Terms*—Direct memory access, Network on chip, Real time system, Time division multiplexing**

I. INTRODUCTION

We are all familiar with the concepts of System on Chip (SoC) and Network on Chip (NoC). SoC is an integrated circuit which integrates different components of a system into a single chip. Traditionally the communication between such components has been done by using bus architectures. But due to the increased demand of communication bandwidth, such bus architectures got replaced by the new concept Network on Chip (NoC). Which is a network used for the communication between different components in a system. The basic architecture of an NoC consists of buffers, routing algorithms and traffic generators. We can design different NoCs by varying all these components to implement different functionalities. Such NoC designs have many area, power and performance tradeoffs with its topology, buffer size, routing algorithms and flow control mechanisms. Here in this work which introduces a NoC architecture which have some advantages like no flow control, buffering and dynamic arbitration.

In this paper, the architecture consist of a router and a network interface. The network interface is used to interconnect the processor core with the network of the system (router). The router is used to interconnect different cores of the system. The network interface used in here is a Direct Memory Access (DMA) controller based one, which is used to initiate the message transfer from the Scratch Pad Memory (SPM) of one core to the scratch pad memory of another core. The hard real time concept of the architecture is ensured by Time Division Multiplexing (TDM) method. The TDM schedule is included in the network interface module and it is simple one and very much area efficient. The router used in here is a asynchronous one, which have three stages and it uses mousetrap latch that works like a clock gating to

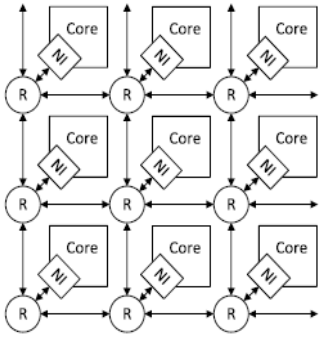

Fig.1 2 dimensional mesh NoC topology

Reduce the energy of the router when it is in a idle state. The basic structure of the architecture consists of a processor, network interface and router.

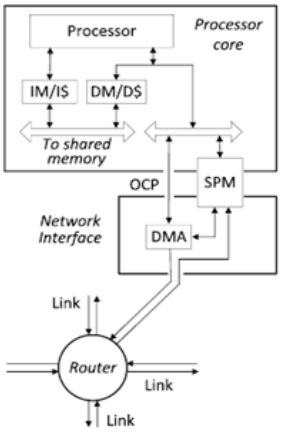

Fig.2 The basic structure of the processor core, network interface and router.

The network interface consists of DMA controller and a five port router. The scratch pad memory acts as both source and target for the message passing between the core.

II. LITERATURE REVIEW

For a system to be hard real time, it should satisfy some requirements like end to end connection and time analyzable. In this paper [5] it discusses a SoCBUS architecture which implements the real time application by using dial up mechanism. Here the connection is established by using some control signals like request and acknowledgement. A wrapper module is used to connect the IP core to the network in this paper.

The Nostrum architecture contributes the service of Guaranteed Bandwidth (GB) along with the existing service of best effort packet delivery [8].The nostrum mesh architecture consists of resources and switches and each switch is connected to its four neighboring switches.

III. PROPOSED SYSTEM

A. Network interface

The TDM schedule used in the architecture avoids buffering, flow-control, and dynamic arbitration. That is, the TDM method will transfer data from one scratch pad memory of one processor core into the scratch pad memory of another processor core without any buffering, flow-control and dynamic arbitration.

The key ideas are to use the dual-ported communication memories (SPMs) for clock domain crossing, and to move the DMA controllers from the processor cores into the NIs. The need for buffering and flow control can be avoided by using this idea and hence allows the DMA controller to deliver the payload data according to the TDM based implementation of the DMA controller. This architecture shown in Fig. 3, is the key contribution of the paper. The key elements of the micro-architecture are the slot counter, slot table, and a DMA table. The slot counter is reset and incremented in all NIs using the same clock and the slot counter defines the slots in the TDM schedule period. A slot is used to define the time taken to transmit a packet into the network.

The slot counter points to the slot table whose entries data consist of a valid bit and address of the DMA table. The valid bit is used to check whether a packet has to be sent in that particular time slot or not. The interfaces denoted M and S are master and slave ports supporting point-to-point read/write transactions. An entry in the DMA table holds all the registers that are found in a normal DMA controller: some control bits (start and done), a read pointer, a write pointer, and a word count. And also it contains an entry, which holds the route of a packet which has to be sent through the network. The reason to split between a slot table and a DMA table is to be able to assign more than one slot in a TDM schedule period to a connection and reserve different amounts of bandwidth for different end-to-end circuits. Storing the route information in the slot table is a special

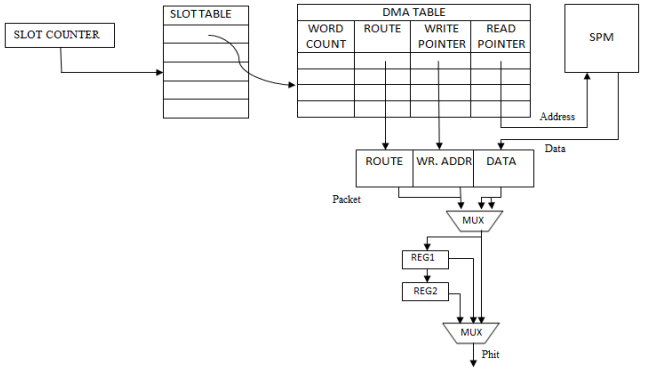

Fig.3 Block diagram of the NI module

Feature in this architecture. This would allow a connection that has been allocated several slots in the TDM period to use different routes in the different slots. Another feature is that, the slot of the network and the DMA table is combined into a single table. This may allow a more efficient hardware implementation, and it is possible if no connection is assigned more than one slot within the TDM period.

The key feature of this architecture is DMA table which can moved from the processor core to the network interface. The TDM schedule is integrated into such DMA table and such table gives specific schedules or time slots for the channel bandwidth guarantee.

DMA transfers have to be interleaved correspondingly, with one DMA controller per outgoing channel. On the other hand, the NI can only introduce one packet at a time into the NoC, and consequently, at a time only one DMA controller can be active. Since each entry of the table refers to one DMA controller, i.e., one outgoing channel a single table based implementation is possible. The dual ported SPMs can be used for clock domain crossing. In this way, explicit clock domain crossings are needed only to program the DMA controllers; the actual transfer of message data does not require any synchronization. The combination of interleaved DMAs and dual-ported SPMs creates a direct path from the SPM to the network of routers, avoiding the need for flow control and buffering in the NIs as well as latency for clock domain crossing for the message data.

Here the Fig.3 depicts the transmitting part of NI with a slot counter, slot table, DMA table, SPM, multiplexers and some registers. The slot counter holds the address of the slot table. The slot table is a memory which consists of a valid data and the address of the DMA table. The valid data in the slot table is used to notify if there is any data to send in that particular time slot, ie if the valid bit is 1, then the address part points to the DMA table else there is no data to send at that particular slot. Now the most important part of the module is DMA table, which is a memory that holds the read address, write address, route information and the word count. The read address is the address of the scratch pad memory, write address is the address of the destination nodes scratch pad memory and word count is used as a data which notifies if the data is send or not to the processor.

B. Router

A generic illustration of a NoC-based multi-processor consists of a set of IP cores and the communication of such IP cores via packet switched structure of routers and links.Typical IP-cores are processors with some amount of local memory, shared memory or IO devices, and the topologies of such NoCs are planar structures like meshes and tree. Each IP-core is connected to the network via a Network Interface (NI) that transates (core-specific) read/write transactions into (NoC specific) packets. Moreover, the NIs typically implements clock domain crossings. Most published NoCs only support best-effort traffic and some published NoCs offer multiple priority-levels in order to support different qualities of service. For hard real-time systems such NoCs are inadequate. In order to provide the time predictability worst case execution time (WCET) should be guaranteed and a NoC for a hard real-time platform must provide some form of end-to-end circuits.

We decided for a TDM-based NoC, since it avoids arbitration, flow control and buffering and thereby reducing the network of routers and links to its minimum pipelined structure of multiplexers, wire segments and registers. Secondly, for an asynchronous router implementation, it avoids the First In First Out (FIFO) synchronizers used in mesochronous designs. Normally asynchronous circuits are larger than their asynchronous counterparts, but in this case the asynchronous router is smaller than the clocked mesochronous router. The use of asynchronous routers in a TDM-based NoC represents a point in the design space that has not been explored before. The TDM router, shown in Fig. 4, is an asynchronous implementation using handshake latches instead of clocked registers. The router is a three-stage pipeline: 1) link traversal; 2) Header Parsing Unit (HPU); and 3) traversal of the Cross bar (Xbar) switch. A router typically has five ports, and this allows the construction of mesh-type topologies. A packet consists of three phits that is, header phit and two payload phits, as shown in Fig.7. The basic unit transits in a pipeline stage is called as phit. Each phit is a 32-bit data word along with three

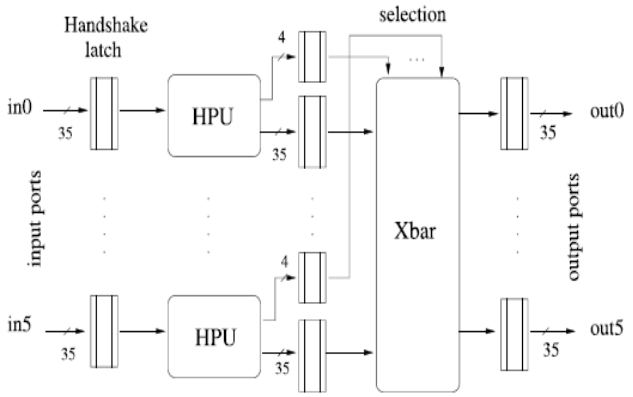

Fig.4 Block diagram of router module.

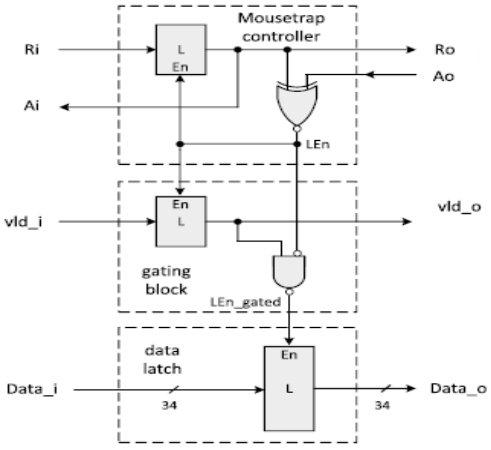

Fig.5 Hand shake latch

Control bits, indicating whether the phit is valid or not (vld), the start of packet, and the end of packet, respectively. The header phit contains the address of the destination SPM and the route of the packet which has to follow. Each input port has an HPU that extracts 2 bits from the route to be decoded to a 4 bit one-hot encoding, selecting the destination output port of the router. At the same time, the HPU shifts two positions of the route field of the header phit to align the header for the next router. The Xbar module will get locked until the two payload phits of the packet has propagated.

In the router module, there are 3 main sub modules, they are hand shake latch, header parsing unit and crossbar switch. The hand shake latch is a module that is used for the power saving purpose by gating each phit in a network. The Fig.5 shows the diagram of the hand shake latch.

The next sub module is the header parsing unit, which is used to decode the header phit in each packet to extract the route information. Each header phit has a 2 bit route information for each router, which is used to select the output port of that particular router. If the phit is a payload phit, then the data is send to the output port which has been selected by its header phit. In case of the transfer of such header phit, the two LSB of the data is right shifted in every router module. The Fig.6 represents the diagram of the header parsing unit.

The router is a 3-stage pipeline as shown in Fig 4. The three pipeline stages are:
(i) link traversal
(ii) header parsing unit (HPU)
(iii) traversal of the crossbar switch (Xbar).

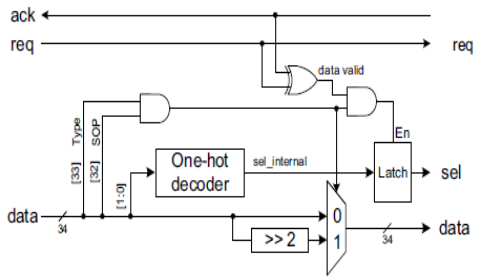

Fig.6 Header parsing unit

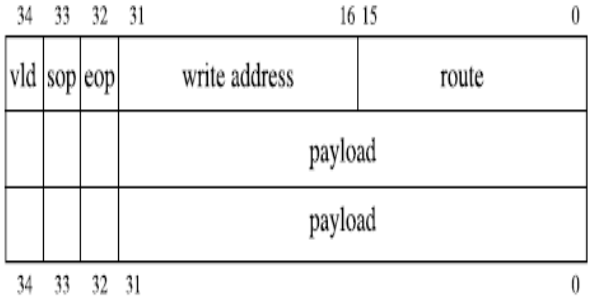

Fig.7 Packet format

As already mentioned we use source routing of packets. A packet is a group of data, which consists of one or more flits and such flits consists of three phits. The first flit in a packet includes header-phit and two data-phits.

The flits count does not affect the router working, but the NI-design in this paper uses packets which consist of a single flit. A phit is 32 bits of data or header supplemented by 2 type-bits: a valid bit that indicates if a phit is present in the current clock cycle and a bit that identifies the header phit. The original aelite-design uses a different encoding of the type bits. A router typically has five ports, and this allows construction of mesh-type topology. The header phit consists of the route and the write address of the destination SPM. Each input port of the router has a HPU that extracts two bits from the route that are used to select the destination output port of the router. At the same time the HPU shifts the two Least Significant Bit (LSB) of the header phit to align the header for the next router along the path. The path through the crossbar is locked until the last phit of the packet has propagated through the crossbar. The format of the packet is shown in the Fig 7. The router module has been simulated and shown in the Fig. 8, which has excluded the request and the acknowledgment part.

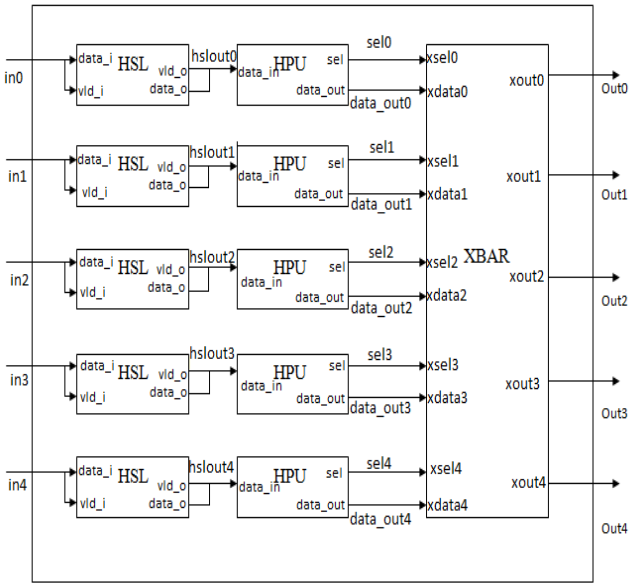

Fig.8 Block diagram of router module

IV. EXPERIMENTAL RESULTS

The modules are modeled and simulated using the language verilog in Xilinx ISE design suite 13.2 .

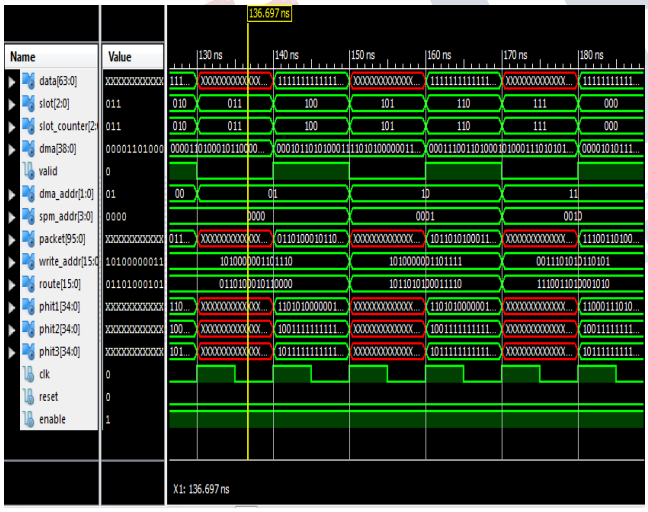

Fig.9 Simulation result of NI with phits

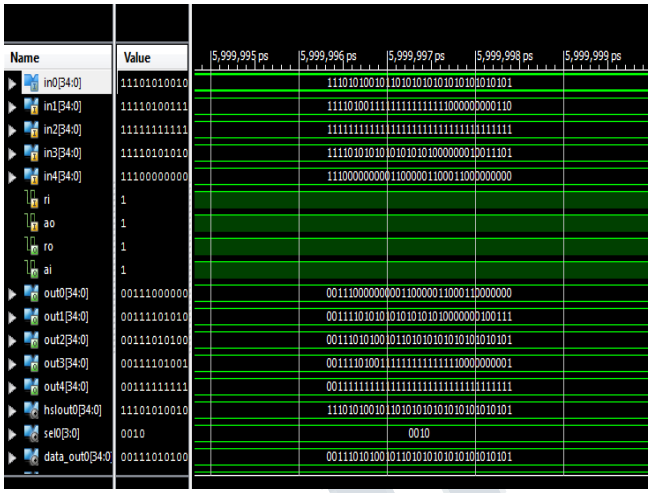

Fig.10 Simulation result of router module

Fig.11 Simulation result of NoC

V. CONCLUSION

This project presented the architecture of NoC; a statically scheduled, time-division multiplexed NoC supporting message passing among processors in a multiprocessor platform optimized for use in hard real-time systems. NoCs are typically designed with a mindset of encapsulation and layering, and resources for buffering, flow control, and clock domain synchronization normally account for a significant fraction of the implementation cost (area). But the architecture mentioned here avoids almost all of this. A novel NI design that integrates DMA controllers and the TDM scheduling, and the use of asynchronous routers instead of clocked mesochronous routers, has resulted in a design in which messages are transferred end-to-end without any buffering, flow control, or synchronization.

In this paper, the design and simulation of the NoC architecture modules like router and network interface has been presented. The NoC architecture is smaller than existing designs with similar functionality. This makes the architecture, a communication solution suited to future multiprocessor systems for hard real-time applications.

REFERENCES

[1] Evangelia Kasapaki, Martin Schoeberl, Member, IEEE, Rasmus Bo Srensen, Student Member, IEEE,Christoph Mller, Kees Goossens, Member, IEEE, and Jens Spars, Member, IEEE,“ Argo: A Real-Time Network-on-Chip Architecture With an Efficient GALS Implementation ”, in IEEE transactions on very large scale integration (VLSI) systems.

[2] Jens Spars, Evangelia Kasapaki and Martin Schoeberl, “An Area-efficient Network Interface for a TDM-based Network-on-Chip”,2013.

[3] K. Goossens and A. Hansson,“The thereal network on chip after ten years: Goals, evolution, lessons, and future”, in Proc. ACM/IEEE Design Autom. Conf. (DAC), Jun. 2010, pp. 306311.

[4] T. Bjerregaard and J. Spars, “ Scheduling discipline for latency and bandwidth guarantees in asynchronous network-on-chip”, in Proc. 11th IEEE Int. Symp. Asynchron. Circuits Syst. (ASYNC) Mar. 2005,pp. 3443.

[5] D. Wiklund and D. Liu“SoCBUS: Switched network on chip for hard real time embedded systems”,in Proc. IEEE Int. Parallel Distrib. Process. Symp. (IPDPS), Apr. 2003, p. 78a.

[6] J. Bainbridge and S. Furber,“Chain: A delay-insensitive chip area interconnect,”, IEEE Micro,vol. 22, no. 5, pp. 1623, Sep./Oct. 2002.

[7] Evangelia Kasapaki, Jens Spars, Rasmus Bo Srensen and Kees Goossens“ Router Designs for an Asynchronous Time-Division-Multiplexed Network-on-Chip”

[8] Mikael Millberg, Erland Nilsson, Rikard Thid, and Axel Jantsch “Guaranteed Bandwidth using Looped Containers in Temporally Disjoint Networks within the Nostrum Network on Chip”.