International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2015): 78.96 | Impact Factor (2015): 6.391

An Efficient Prototype for Gals Systems in Asynchronous Network-on-Chips through Multiclocking

P. Vijayalakshmi[1], K. Sathiya[2]

[1]A.P (Prince Shri Venkateshwara Padmavathy Engineering College), Chennai

[2]A.P (Prince Shri Venkateshwara Padmavathy Engineering College), Chennai

Abstract: ***GALS archetype is to partition a system design in decoupled Clock- independent modules. GALS design eliminates the issue of using global clock and decreases area overhead when compared to purely synchronous or purely asynchronous designs. Network-on-chips vigorously benefit to such a globally asynchronous design methodology. Clock free interconnect networks improve authenticity by removing clock-domain passing synchronizations and by using delay-insensitive mediators for solving routing issues. This paper presents an trendsetting methodology for network-on chip Globally- Asynchronous Locally- Synchronous (GALS) system paradigm. High performance multi- clock FPGAs are overburdened for easy and fast prototyping of GALS systems based of an Asynchronous Network-on-Chip (ANoC). Modularity property of asynchronous circuits is fully oppressed to design regular distributed link topologies by the means of basic topology-free building blocks, with a focus and special design effort to solve mediators and synchronization problems. The design is carried out using FPGA and simulation and synthesis reports are shown.***

Keywords: GALS - Globaly asynchronous locally synchronous

1. Introduction

The world of IC is becoming more complex. The decrease transistor dimension has lead to deep submicron problems like quantum effects and wire delays. These DSM problems make designing and verifying chips far from trivial. For every new design these problems need to be solved, making design effort increase exponentially with decreasing transistor dimensions. Usually a chip requires bus type structure to link all functional units. But a bus is designed to handle a small (finite) number of functional units. Effort has been put in to finding solutions to this problem for instance by creating multiple bus structures with bridges in between. But these solutions don't scale well and come with their own set of drawbacks. To solve these problems the Network-on-Chip (NoC) concept has been introduced. The main idea behind NoC is to formally divide the design into functional units and connect these units via an universal communication Network.

Many of today's VLSI design are based on globally synchronous design. The new SOC design faces the challenge of distributing a high- speed low-skew local clock in a large die. Proper clock distribution needs numerous buffers and a carefully designed clock tree which introduces considerable area and power overhead. Asynchronous design methodologies can eliminate such overheads by removing the clock signal from design. The properly designed asynchronous systems consume less power and may be faster than the synchronous counterparts. But the price to pay is inclusion of widespread control overhead. But this often increases the die size and leads to complication in design. Also the lack of reliable tools for asynchronous circuits makes it even more unacceptable for asynchronous designs to substitute the current synchronous VLSI design methods.

The Globally Asynchronous and Locally Synchronous is combination of both synchronous and asynchronous designs which combines the advantage of synchronous and asynchronous design. GALS have emerged to solve distribution problem and offer low power while allowing synchronous design benefits. GALS generally involves in portioning of large system in to number of independent synchronous modules. Each module is allowed to run from its own local clock. Data exchange between any two locally synchronous modules strictly follows a full handshake protocol.

GALS design approaches incorporate a combination of the advantages of both synchronous and asynchronous design methodologies while avoiding their disadvantages. .The major reason for pursuing GALS design methodologies are;

- Easy to design systems with multiple clock domains.
- No global clock signal to distribute.
- Power efficient: System operates only when data are available to form of clock-gating
- Limited, standard asynchronous circuitry.
- Comparable performances.

The issues to be addressed during the design of a GALS system are:

- Metastability avoidance.
- Latency.
- Flow control.
- Local clock alteration.

The design of GALS starts with partitioning of fully designed and tested synchronous circuit. Availability of large number of verified synchronous IP block is one of the motivations behind this approach. The synchronous circuit should be partitioned into independent locally synchronous island. This process includes partitioning data path and

decomposing the central controller to generate autonomous synchronous islands. While LSIs perform their internal tasks synchronously, they must be equipped with asynchronous wrappers to be able to participate in globally asynchronous communication.

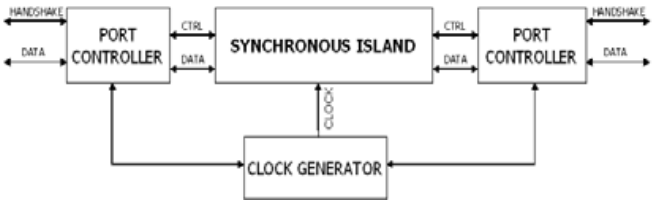

Figure 1: General GALS Module

The wrapper contains a local clock generator and asynchronous input and output controllers to communicate with other GALS circuit design. The local clock generator is made plausible clock and controlled such as to prevent any timing violation from occurring within the locally synchronous island's data interface. The problem of metastability is thus addressed by the ability to suspend the local clock whenever data and sampling clock edge occur too close to each other.

A Port controller is in charge of managing all data transfers on a particular port in a GALS system. It gets enabled by the synchronous island and has to synchronize data transmission on the GALS data transfer channel and local clock phases. All transfer channels employ a rendezvous scheme i.e., the transfer takes place at one point in time when both sender and receiver do enable the transmission. In order to transmit data fast and efficiently the port controllers need to act independently from the local clock signal. This is achieved by implementing them as asynchronous finite state machine. The methodology for designing ANoCs is focused on solving synchronization problems. The two major synchronization bolts for a GALS system are synchronization at clock domain boundaries and arbitration between concurrent requests. Such circuits have a non- deterministic behavior.

2. Requirement of NoC Architecture

Most of the circuits are based on synchronous Network on chip architecture. Since the components are not active most of time, the asynchronous design will be useful for reduction in power. This paper deals with network on chip based on asynchronous approach. By doing this we can achieve 20 to 30% of power reduction.

3. Implementation of NoC

Our methodology for designing ANoCs is focused on solving synchronization problems. The two major synchronization bolts for a GALS system are: synchronization at clock domain boundaries and arbitration between concurrent requests. Such circuits have a nondeterministic behavior. We put special invest and design effort to improve reliability/performance tradeoffs of these synchronizer circuits.

As discussed in the introduction using an ANoC is in itself a reliability improvement by removing clock-domain crossing synchronizations through the interconnect network. However *clocked* synchronizers are still required between Synchronous peripheral Blocks (SB) and the ANoC.

Arbitration circuits, or simply arbiters, are required where a restricted number of resources are allocated to different user or client processes. Packet routers are such a case. In the case of an asynchronous NoC, delay-insensitive arbiters have this main advantage of being hundred-percent beliable (enough time is given to Resolve metastability). Reliability of on chip communication systems is becoming a major issue since the increase transaction rates is drastically reducing the so-called Mean Time before Failure characterizing clocked synchronizers.

3.1 Modular Asynchronous NoC Structure

The interconnect network fully exploits the modularity property of asynchronous circuits. The construction of asynchronous NOCs has been divided into five basic components (figure 2 and 3). They are

- Wrapper Adaptor
- Synchronization and performance interface
- Packet transport
- Parallel request sampling priority arbiter
- Packet routing

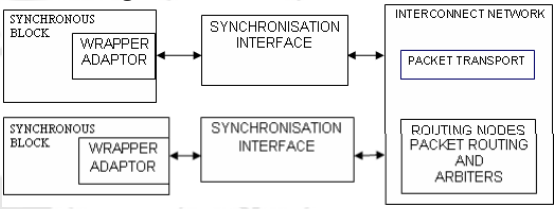

Figure 2: Asynchronous NoC Centric GALS Architecture

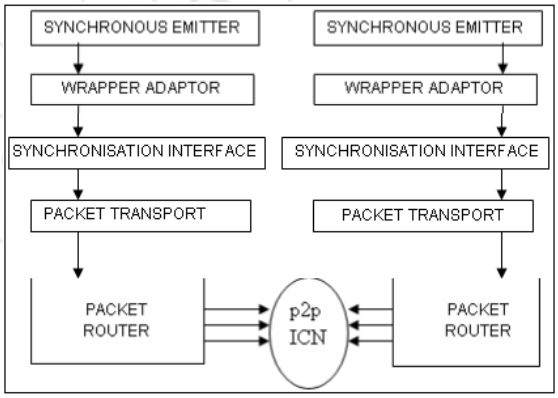

Figure 3: Layer Structure

3.1.1 Wrapper Adaptor

This resource is required to translate between the communication protocol used by a synchronous and asynchronous peripheral and interconnect network. The WA component adapts both flit and packet levels of the communication protocols. In the case of asynchronous peripheral blocks WA is optional. The Wrapper circuit consists of input port controller, output port controller, local clock generator. The input/output port controllers may be of either demand type or poll type. The local clock generator is made of mutual exclusion element.

3.1.2 Synchronization and Performance Interface

This component binds the synchronous peripheral block clock frequency domain with ANOC using FIFO decoupling method. The SPI is built of standard double flip-flop synchronizers and of and asynchronous FIFO. The asynchronous FIFO transforms synchronous protocol into corresponding asynchronous one. The DFF resynchronize asynchronous signal with SB clock. The DFF offers very sufficient reliability for two clock cycle latency per input sampling compared to numerous clocked synchronizers' improvement. The asynchronous FIFO adapts relative speed between the SB and the ANoC. For AB such a FIFO is optional and can be used for pipeline performance optimization. The level of parallelism between data and control flows is improved and two versions are delivered such as low latency or low power consumption according to design requirements. The level of parallelism between data and control flows is improved and two versions are delivered such as low latency or low power consumption according to design requirements.

The level of parallelism between data and control flows is improved and two versions are delivered such as low latency or low power consumption according to design requirements. The level of parallelism between data and control flows is improved and two versions are delivered such as low latency or low power consumption according to design requirements. The level of parallelism between data and control flows is improved and two versions are delivered such as low latency or low power consumption according to design requirements. The level of parallelism between data and control flows is improved and two versions are delivered such as low latency or low power consumption according to design requirements.

3.1.3 Packet Transport

This resource adapts the physical or signal level of communication protocol. The PT component provides successive protocol conversions from SPI component to delay- insensitive NoC core for best power consumption and robustness. Between SPI and PR layers, bundle data protocols are converted in QDI protocols for better robustness. Between the packet routers, the four-phase protocols for long interconnect links for better power consumption.

3.1.4 Parallel-Request Sampling Priority Arbiter

This resource provides a self-timed arbiter with a decoupled arbitration process and a 100% reliable request sampling structure is based on delay-insensitive synchronizers. One of the critical components of multiclock GALS is the communication system. Such an on chip communication has to be flexible to interface in- house and external virtual component, providing high bandwidth, low power consumption, arbitration scheme and routing capabilities. Several SBs were running concurrently may require same resource which may lead to contention problems. In this case arbiter is needed to solve conflicts and to ensure that only two SBS are communicating at a time. These algorithms can be classified according to their hardware characteristics. The priority arbiters are basically classified as

- Static priority arbiters
- Dynamic priority arbiters

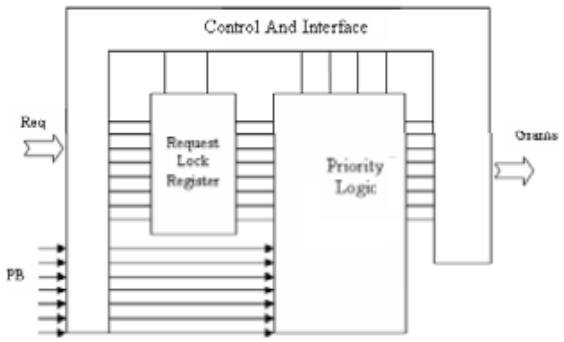

Figure 4: Static Priority Arbiter

In Static priority the fixation of request vector is completely separated from the combinational circuit realizing the priority function. In order to be considered for the next grant, one or several requests must arrive before the moment of request vector fixation (figure 4). This happens either if several requests arrive simultaneously or if these arrive or stay pending during the critical section in one of the clients. The priority function can be changed by modifying the combinational circuit without altering the arbiter structure. Further this can be done at initialization time by setting up additional inputs of the priority circuit, thus making the arbiter programmable.

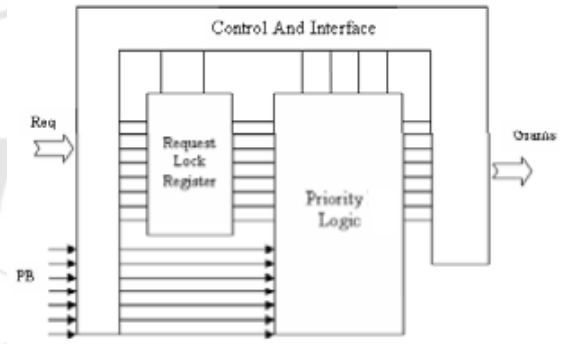

Figure 5: Dynamic Priority Arbiter

A Dynamic priority arbiter performs comparison of the priorities supplied with each request and grants the one with the highest value (figure 5). The priority logic in the dynamic case has a tree structure in which the control flow is maximally decoupled from the data-path by means of an early propagation of the 'valid'-'invalid' signal, concurrently with the processing the priority data. This leads to significant reduction in the overall arbitration delay when the number of active requests is low.

3.1.5 Packet Router

This resource offers a modular routing of data items for the transaction services i.e. packet level services such as burst mode or split transactions. In the figure 2 PRS-PA and PR are put together because they are parts of ANoC routing nodes

3.2 Switches Architecture of ANOC Routing Nodes

Packet router is the core component of an interconnect network. The packet routers are assembled with modular elementary block with the objective of low complexity and low power consumption.

3.2.1 Emitter Module

Emitter module consists on n-to-1 switch around the packet router and priority arbiter (figure 6). The packet router resource is decomposed in tothree modules. They are

- Packet analyzer
- Data path controller
- MUX module

The emitter component delivers two major classes of packet level services such as

- Arbitration service
- Transaction service

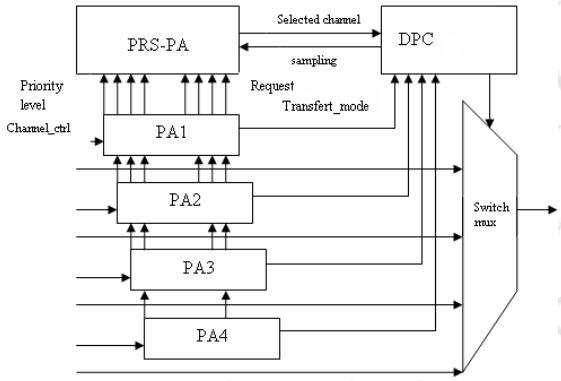

Figure 6: N-To-1 Switch

The PA block decodes the channeli_ctrl message in order to extract arbitration and transaction information and to drive it respectively to PRS-PA and DPC modules. Arbitration information is composed of request and priority level channels used by the PRS-PA module to arbiter the incoming requests. Once a channeli_data is elected, PRS-PA informs the data path controller module through selected_channel. DPC exploits it and the transfert_mode channel to control data flow on the elected channneli_data and to drive the switch output i.e. mux module. Through transfert_mode channel, transaction information delivers packet status such as single flit or for burst mode, start packet flit, body flit, end of packet flit. Once the packet transfer is achieved, DPC module informs the sleeping arbiter module PRS-PA through sampling _channel that a new transaction can start.

3.3.2 Receiver Module

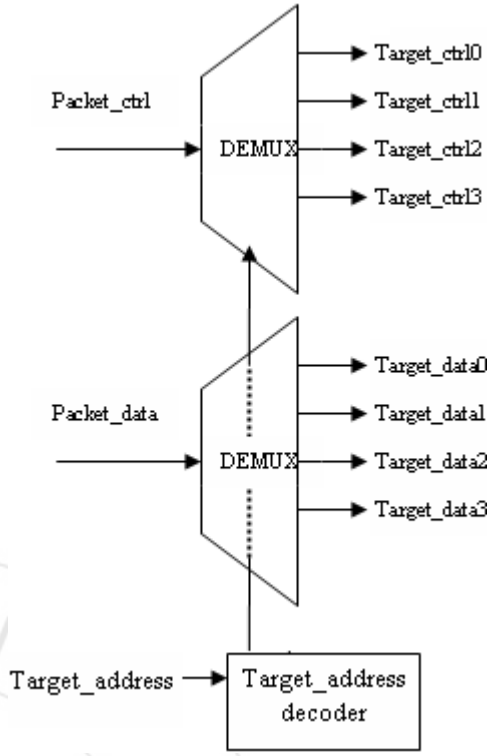

Figure 7: 1-To-M Switch

The 1-to-m switch, or receiver is PR component which realizes the dual operation by driving the input i.e. packet_ctrl and packet_data_channels to the selected target address. By composing these switches we can build in short design time fast and efficient routing nodes

4. Case Study & Simulation Result

The design of the ANoC is done using Hardware description language. The simulation of VHDL is done using Modelsim and synthesis to be carried out using Xilinx Project Navigator. The simulation of ANoC is shown below. A Case study is implemented using two processors and RAM using Asynchronous Network On Chip. Processors are operated in different clock frequencies. The two processor shares one RAM resource.

The processors performs memory read and write operation using ANOC. Whenever two processors wants to access RAM then the processors generates request and priority signal to ANOC. The parallel request sampling dynamic priority arbiter in ANOC perform priority comparison among two processors and grants the resource to the corresponding processor depending upon the priority algorithm.

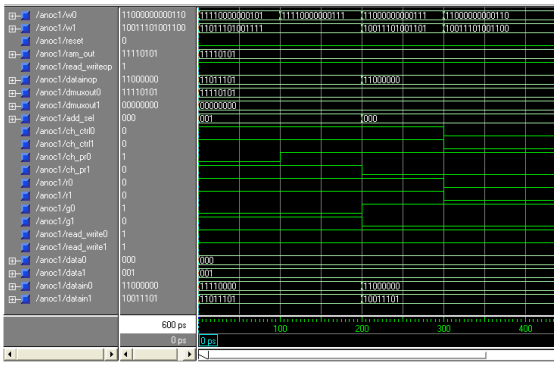

Figure 8: Simulation result of ANoC

Here in this project two level priority is used. When only one processor raises the request then without any delay the resource is granted to that particular processor. Only when two processor raises the request then it enters in to the priority algorithm. The simulation of case study is shown below.

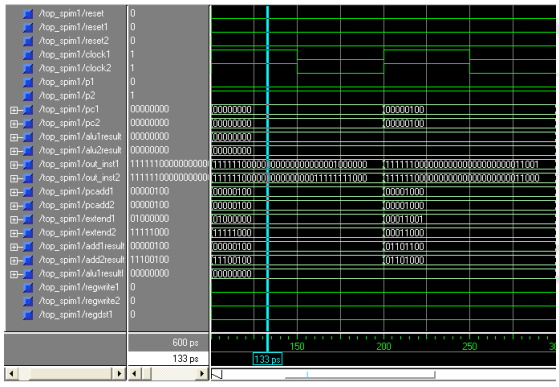

Figure 10: Simulation Result of Case Study

5. Conclusion

In this paper, it is shown that a GALS system with mixed synchronous and asynchronous circuits can be fast prototyped onto a multi-clock FPGA thanks to an asynchronous NoC which releases global design constraints. This ANoC properly interfaces communication protocols with standard synchronous peripherals and offers a reliable and efficient arbitration solution. Strongly modular building of ANoCs ensures short design time and consequently a fast GALS system prototyping. The interconnect topology g enerator delivers several configurable interconnect topologies which facilitate the system architecture exploration. This design will give communication performances for asynchronous NoC c apabilities to deliver fast and robust communication quality of service. In future work will be to improve ANoC fast design and mapping, especially for Wrapper Adaptors which are the bottleneck of the communication performances. Prospective works will be focused on to deliver a dedicated synthesis tool for fully automated asynchronous regular interconnect topologies generation.

References

[1] Jerome Quartana, Salim Renane, Arnaud Baixas,Laurent Fesquet,Marc Renaudin, ”GALS Prototyping Using Multiclock FPGAs And Asynchronous Network-On-Chips” IEEE 2005.

[2] J. B. Rigaud, J. Quartana, L. Fesquet et M. Renaudin, "Modeling and design of asynchronous priority arbiters for on- chip communication systems", Proc. of the VLSISOC'01, 11th IFIP Int. Conf. on Very Large Scale Integration, Montpellier, France, 3-5 Dec. 2001.

[3] E. Beigné, F. Clermidy, P. Vivet, A. Clouard, M. Renaudin,"An Asynchronous NOC Architecture Providing Low Latency Service and its Multi-Level Design Flow", 11th IEEE Int. Symp. on Asynchronous Circuits and Systems (ASYNC'05), March 14-16, NY, USA, pp. 54-63, 2005.

[4] T. Q. Ho, J. B. Rigaud, M. Renaudin, L. Fesquet et R.Rolland, "Implementing Asynchronous Circuits on LUT Based FPGAs", *Proc*. of the Field-Programmable Logic and Applications, Reconfigurable Computing Is Going Mainstream, 12th Int.Conference, FPL 2002, Montpellier,France, September 2- 2002.