International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-9 Issue-4, February 2020

Latency Minimization using Mesochronous Scheduling in MPSoC Operation

Sukanya.K, G. Laxminarayana

Abstract: ***High speed computing is the upcoming challenge for next generation applications. To cope with high speed operations, new processing architectures are evolving. Multi processor design is one optimal design approach for such need. In the design development of multi processor unit, Multi-Processor System-on-Chip (MPSoC) has an outcome in the domain of VLSI design. MPSoC are designed to process multiple instructions and data handling simultaneously. The parallel processing feature make this unit faster and optimal design for upcoming applications. However, MPSoC operations have a latency issue in clock allocation and resource utilization, which effects the processing efficiency and introduces delay and resource overhead in MPSoC interface. This paper outlines a Mesochronous operation in MPSoC design for minimizing latency in clock allocation and resource allocation, hence improving the speed of operation.***

Keywords: ***Latency minimization, MPSoC design, Mesochronous operation, VLSI design.***

I. INTRODUCTION

Multi-Processor System-on-Chips (MPSoC) are the major developing architecture in the area of fast computing interface. With the evolution of new technology, the need for MPSoC has emerged. However, the computing overhead and power requirement for such complex architecture has resulted in its optimization requirement. To develop the solution of this constraint, designers deal the issue in two modes where the designer customize the design to the application limit [1] and by scaling the operation to the limited voltage/frequency operation [2,3]. Wherein, customization is observed to be an optimal approach, the design overhead is considerably high [4]. In the optimization of power and processing overhead, the design approach involves in monitoring the communication protocol and the signal interfacing among different units [5] in the processor unit. The diversity in the design units, and the components used in such design are also a major constraint in the optimization process of a MPSoC unit [6]. The optimization constraints also threshold to the operable frequency and system throughput [7] in terms of different applications. Therefore the design approach is defined with an internal updation process in clock allocation [8] and FIFO based approach for synchronization in sub unit operations among multiple processing units.

Revised Manuscript Received on February 28, 2020.
***** Correspondence Author
Sukanya.K*, Department of E.C.E, TKR College of Engineering and Technology, Ranga Reddy, Telangana, India. sukanya.addagatla@gmail.com
G. Laxminarayana Department of E.C.E, Anurag College of Engineering, Ranga Reddy, Telangana, India. gln9855@gmail.com

Here each of the core unit is interconnected for a synchronization of data exchange among different core units [9]. Each of the processing blocks called IP core uses dual clock FIFO architecture. However, if all IP blocks use dual-clock FIFO architecture for a single common purpose, the resource has a higher risk than provisioning because all IP interfaces need to be configured conservatively, as each IP core has different speeds and throughput need [10]. In a Worst case scenario, for example, the buffering of such synchronization parameter should be adjusted based on the source and receiver frequency comparison [11]. In addition, the descriptive presence of information for the frequency ratio (such as, the on-chip interconnects works at faster speed than interlinked IP units), coupled with information of performance limitations, can lead to twin high impact specializations [12]. Hence, the dual clock FIFO architecture has a finally large area and power conservation. Various usage of such coding is presented in [12-17]. Because the designs do not use clocks, it is difficult in accepting the synchronization operation across two variation of clock [18]. Among the synchronization of the suggestive core units, the delay factor in clock system is unaddressed. This limits the synchronization in such approach. The past developments illustrate an effort in minimizing the delay factor in MPSoC design. However, the latency due to resource allocation at the time of processing is not overcome. Here each of the instruction process has a delay in processing clock; it needs to allocate the clock delay and the transition of this allocation results in processing delay to the system. In this paper, a new latency control mechanism by introducing enhance clock library function is presented. This approach reduces the interchanging data and instruction delay and overcome the latency overhead in instruction execution. To present the paper, this paper is outlined in 6 sections. Section 2 presents the approach of library coding in MPSoC design. Section 3 presents the proposed approach of latency monitoring a library coding for Mesochronous operation. Section 4 presented simulation result for the developed approach with conclusion presented in section 5.

II. DISTRIBUTION OPERATION IN VA-MPSOC [19]

CoreVA-MPSoC presents the target application for a hierarchical interconnects architecture which is embedded and energy-efficient systems. In this CPU clusters are well connected to many core CPUs in the cluster which share the similar address space in the core. At its original configuration, each CPU can read and write local data of other CPUs using a bus-based interconnect.

To prevent cycles for penalty operations, the CPU is connected to a cluster interconnection via FIFO buffer. Standard data bus width topology is configured at the time of design of such processor unit.

For example, Advanced Microcontroller Bus Architecture (AMBA) uses 32 bit or 64 bit data bus width for the AXI4 Interconnect standard. AXI4 defines address and data transfer for specific addresses. In addition, separate reading and writing channels allow parallel read/write (R/W) requests for a common bus. Steps for register allocation at interconnect to facilitate space and route duration compensation and increase the frequency to maximum of MPSoC clocks can be added. Here, the architecture does not support the best read requests because it does not run in the order of execution for all cores. The CPU has the lowest CPU latency of 4 clock cycle of operation. A total of five intermediaries (1 per channel) are required for Share Bus implementation, and each operation (Read/Write) requires two intermediaries for crossbar interconnections. During the two CPUs communication, Network Interface (NI) through Network-on-Chip (NoC) interface is developed. Here, each CPU cluster is located by a distinct X and Y coordinate index in a 2D-topology. A shared address space in the cluster is used for all memories and units in the cluster. NI bridge address-based communication during CPU cluster and packet exchange using a router interconnects. As such, it provides capable flow control in the CPU core when reducing CPU time of operation for this contact. To do this, packet data is stored and read directly from all CPU's local memory. Therefore, CPUs take advantage of local memory access delays in their operation. In addition, NI works as a DMA controller by distributing and getting parallel data to the CPU. In the original configuration, NI is coupled to the Cluster via a Master and a Slave Port. Packets may be sent to different channels for the separation R/W transactions. Where, while writing a data at the same time, the AXI master port of NI can be sent.

An effective communication approach is needed in the exchange of schedule task for different CPU interface. CoreVA-MPSOC [19] uses a communication model with integration of a unidirectional communication channel. This approach offers more scalability and efficiency than shared memory concepts, where memory accessibility can be interrupted. Generally, one task reads from one or more input channels and writes it to one or more output channels. Each channel controls one or more R/W data storage. Synchronization is governed by buffer size granularity. If buffer requests from a channel, the CPU does not get enough data or a free basic buffer is not available to write.

However, as the data is received from a channel, the application accesses any memory location in that buffer randomly. In addition, no further synchronization is necessary. Here, on the completion of a task, the CPU interacts with the other units to exchange the status and allocate the register for reuse. The resource sharing hence reduces the overhead. However the allocation delay is considerably large. The delay due to resource allocation and synchronization is also considerable, where the computation of time delay is constraint to data interface among different CPU units using NoC interface. The NoC here acts as a bus arbiter as routing bus for data exchange. A considerable delay is observed in this exchange leading to decrease in processing speed. Hence, the delay constraint is to be minimized, which is focused in this paper. To develop a faster clock allocation in synchronization process a new delay mapping operation using time stamp library is proposed.

III. MESOCHRONOUS CLOCK VIRTUALIZATION IN MPSOC (VR-MPSOC)

In the proposed approach of virtualization of MPSoC operation, a virtual delay computation unit is presented. This proposed approach modifies the present architecture of MPSoC unit with each core unit provided with a library unit; each of the library unit is defined with a pre-computed delay parameter for each operational instruction of processor unit. In this design approach, the instruction set for a multi-core processor unit is referred, where the instruction is categorized as 1, 2 or 3 byte instruction. Each of the operation here is performed as direct addressing operation or one operand indirect addressing operation or two operand indirect addressing operation. For each of this type, the clock latency is computed. The computation of the delay is measured as an aggregated delay of the physical delay due to device manufacturing and the aggregated delay of the setup and hold delay time for the data. Each of this delay is computed for each type of instruction and set on to each core unit as a library. This process of delay entry is performed at the design process, as the processing instruction to a design processor is constant, for each of the instruction the delay parameter is constant. Here, each of the instruction is processed for total delay computation and each instruction is set with a corresponding delay in the library function. Each of the libraries is buffered as synchronization mapping table in the core unit. In the operational phase, this library unit is mapped with the instruction to process and the delay is mapped to the arbiter unit. The arbiter is defined at NoC interface of MPSoC which perform the bus arbitration process. Each of the bus line is allocated based on the core bus request generated from the core unit. In this process of allocation, for each allocation the delay of the mapped instruction is applied. This reduces the delay computation overhead and leads to minimization of latency in MPSoC operation. The latency parameter of the proposed approach is the aggregated delay due to clock allocation and computation delay. In each of the instruction execution, the data or instruction need a time of buffering of instruction/data to synchronize the operation. Each of this unit is processed for a delay value as per the type of the instruction. As the aggregated latency is the total time of switching delay from allocation to computation operation and the delay time period for the clock to synchronize and the delay is large. In addition to the processing delay, a routing delay is observed during the bus allocation and data exchange. Wherein efforts are made to reduce the routing delay by an optimal floor planning of the architecture, the delay is considerable in operation phase.

To overcome this delay, the major factor of switching and computation is eliminated by the library unit. This is a virtual realization of the time stamp unit, which returns the required delay value in operational phase.

This leads to the development of virtual MPSoC design termed as 'Vr-MPSoC' unit. The proposed approach is summarized in the following algorithm presented below.

Algorithm (Clock switch virtualization)

Process Initialize:
Step 1: *define the cluster of CPU*
Step 2: *allocate the arbiter for data and instruction*
Step 3: *allocate the operation instruction for each CPU*
Process read:
Step 1: *generate a read offset signal to library latency*
Step 2: *recover the time stamp for each instruction*
Step 3: *record the delay to offset library*
Processes execute:
Step 1: *Read instruction*
Step 2: *Decode instruction type*
Step 3: *Read offset value*
Step 4: *Allocate to data and instruction register*
Step 5: *Read data*
Step 6: *Execute instruction*
Step 7: *Write back*
End

The operational block diagram for the proposed approach is presented in Fig. 1 below.

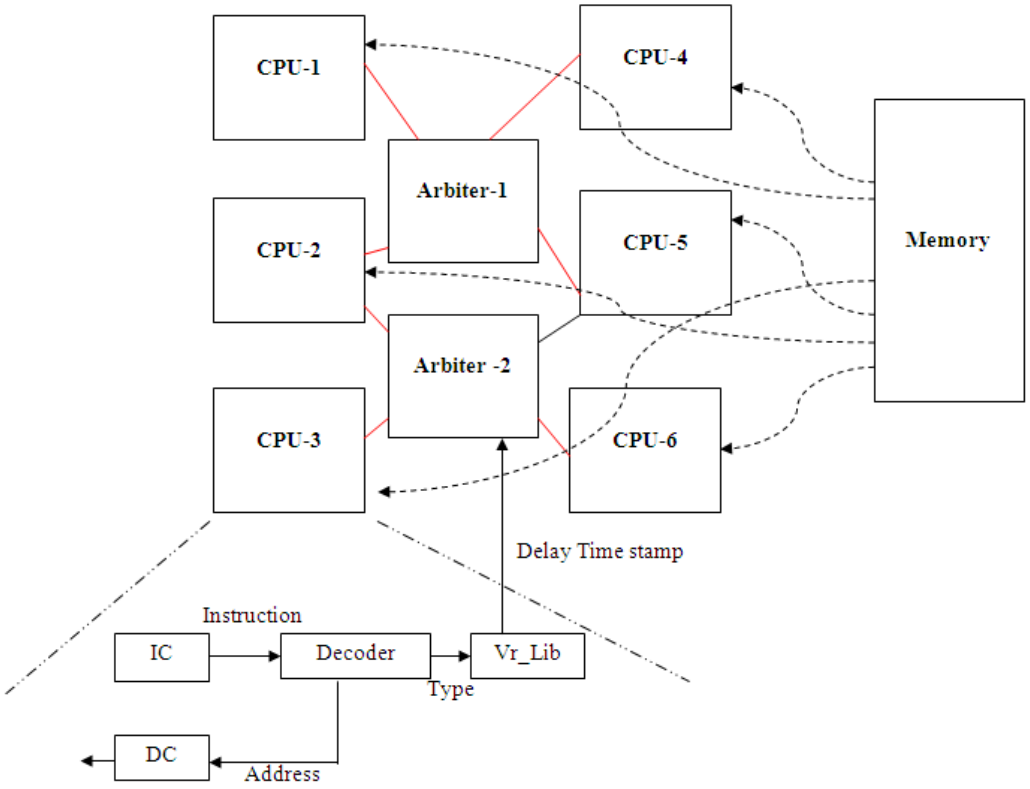

Fig. 1: System Architecture for the proposed MPSoC interface

The processor unit is processed in two phase of operation, where phase-1 process the update phase, where each of the decoded instruction are mapped to the library unit. For each of the instruction type decoded a delay time stamp is allocated to the arbiter, which intern allocates the delay value based on the clock cycle during execution process. The computation and allocation process is eliminated and the arbitration operation is performed based on the scalar time delay value. This results in low latency to the system.

IV. SIMULATION RESULTS

This proposed work is validated in three simulation levels, wherein the operational functionality of the proposed approach is measured with respect to the timing operation of the proposed work. The latency in allocation and operational overhead is computed. In second level of realization, the proposed approach is validated for implementation using Xilinx ISE synthesizer targeting to FPGA device. This result illustrated the processing speed, area, latency and system throughput. The third part of the simulation result is performed as an analysis of different instruction density.

A) Operational functional verification

The HDL definition of simulated specific work has been developed for time monitoring in the Aldec tool. To implement the proposed approach, the developed design is aimed at Xilinx FPGA devices. Power, delay, throughput and area are evaluated by measurements. The results are explained below. Fig. 2 illustrates the set of instruction used for the test validation of the proposed work.

This approach performs different functional operation by the usage of set of instruction buffered as instruction cache in the CPU core unit. Each of the core unit is buffered with the processing instruction and the data fetched from the main memory.

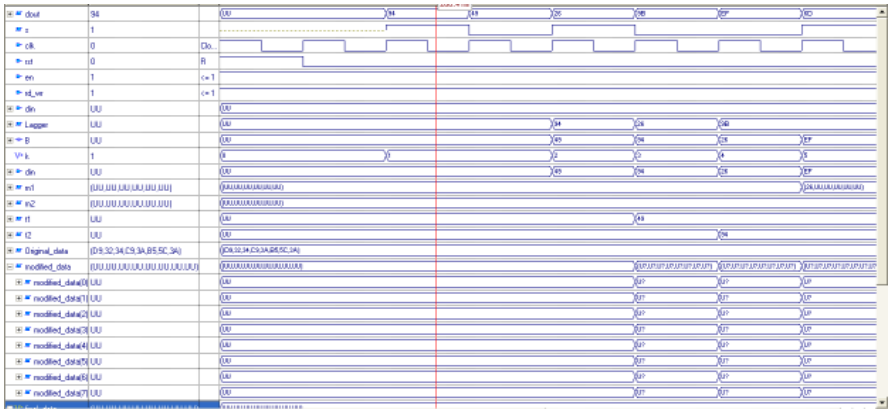

Fig. 2: Operational instruction used for testing

In the test process, each of the instruction is passed to the arbiter unit, where the instructions are mapped with the delay constraint as illustrated in Fig. 3.

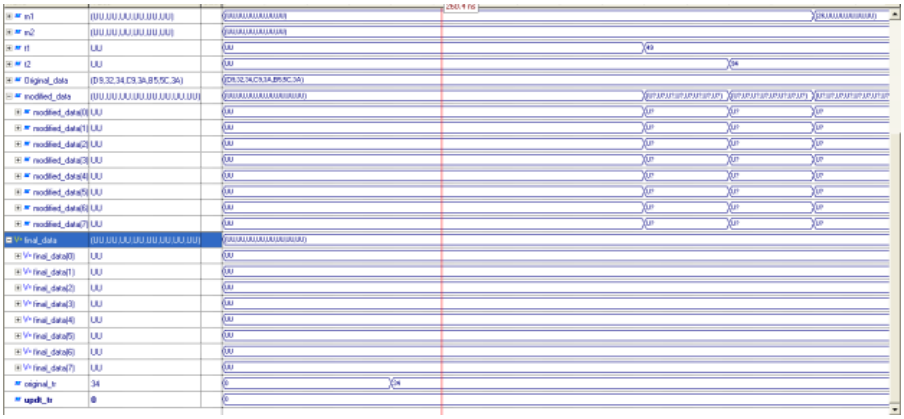

Fig. 3: Mapped instruction of delay metric at arbiter unit

For each of the instruction, the mapped clock pulse result in a new data formed which is decoded as delay instruction set at the arbiter. During the execution, each of the register is allocated with the delay value for input and output clock delay in the processor unit. For each of the instruction the mapped clock pulse result in a new data formed which is decoded as delay instruction set at the arbiter. Each of this processing unit, while performing an execution read, the delay is directly allocated during processing. The process of delay mapping and allocation is illustrated in Fig. 4.

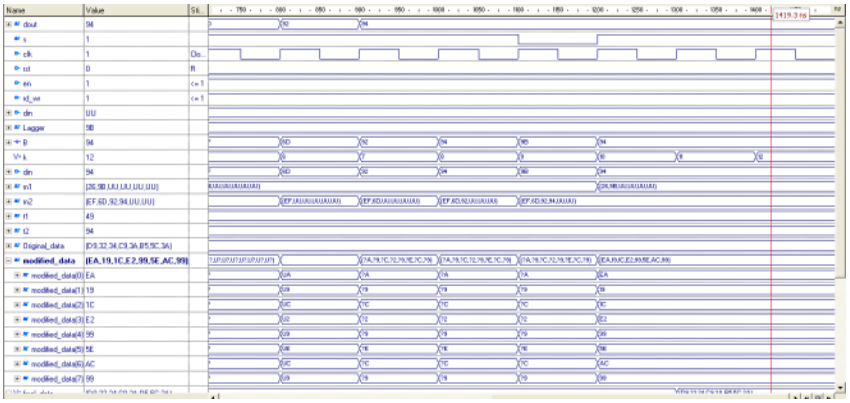

Fig. 4: Delay instruction allocation at arbiter unit

In the process of instruction execution for a 4 set of instruction, the latency for the proposed approach is observed to be 49 compared to 54 for VA-MPSoC design. The observation is illustrated in Fig. 5 below.

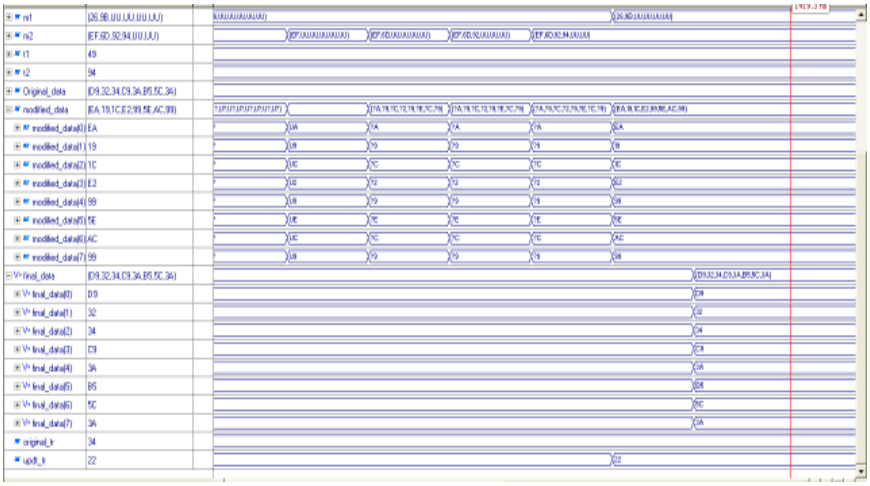

Fig. 5: Latency measurement for the developed system

B) Implementation result

The implementation of the developed approach is targeted to Xilinx FPGA device for a Spartan family. The implementation report obtained is presented in Fig. 6.

TRB Partition Summary

No partition information was found.

Device Utilization Summary

Logic Utilization	Used	Available	Utilization	Note(s)
Number of Slice Flip Flops	113	88,192	1%	
Number of 4 input LUTs	123	88,192	1%	
Logic Distribution				
Number of occupied Slices	134	44,096	1%	
Number of Slices containing only related logic	134	134	100%	
Number of Slices containing unrelated logic	0	134	0%	
Total Number of 4 input LUTs	261	88,192	1%	
Number used as logic	123			
Number used as a route-thru	138			
Number of bonded IOBs	34	1,164	2%	
IOB Flip Flops	14			
Number of PPC405s	0	2	0%	
Number of GCLKs	1	16	6%	
Number of GTs	0	20	0%	
Number of GT10s	0	0	0%	
Total equivalent gate count for design	2,633			
Additional JTAG gate count for IOBs	1,632			

Synthesis Report

Fig.6: Report of Xilinx FPGA implementation for the developed system

A Power Analysis of the implemented design is computed using X-power analyzer of Xilinx tool. A power rating of 181mW of power rating is obtained.

```
Power summary:                                I(mA)      P(mW)
----------------------------------------------------------------
Total estimated power consumption:                         181
                                               ---
                          Vccint 1.50V:          85       128
                          Vccaux 2.50V:          20        50
                          Vcco25 2.50V:           2         4
                                               ---
                               Clocks:            0         0
                               Inputs:            0         0
                                Logic:            0         0
                              Outputs:
                                 Vcco25           0         0
                              Signals:            0         0
                                               ---
                 Quiescent Vccint  1.50V:        85       128
                 Quiescent Vccaux  2.50V:        20        50
                 Quiescent Vcco25  2.50V:         2         4
```

Fig. 7: X-power report for developed system

The timing report for the developed system illustrated a maximum operating frequency of 129.98MHz with a time period of 7.6ns. The device has a setup delay of 2.9ns and a hold delay of 3.6ns. The placement of the logical design with routing and area coverage is observed using Xilinx-route and place operation. The Interconnection for the developed approach is shown in Fig. 8.

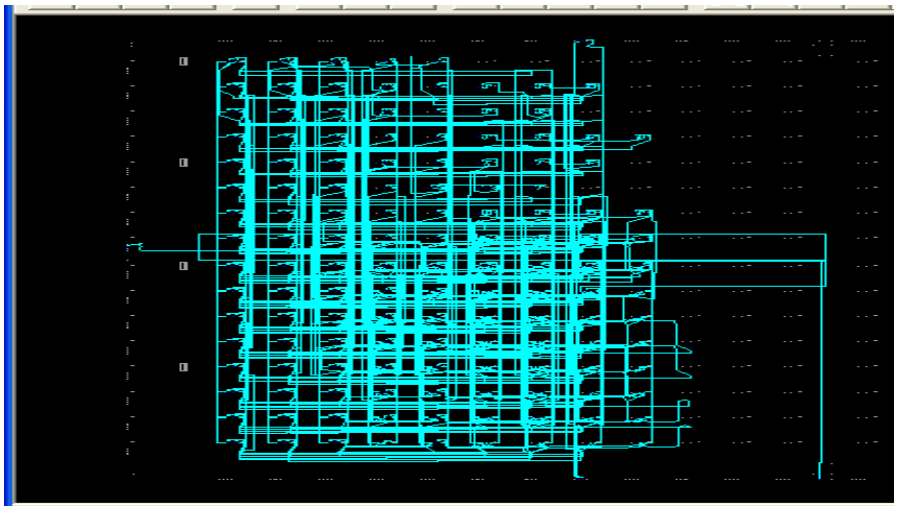

Fig. 8: Logical interconnect of CLB in targeted FPGA device

The logical placement of CLB unit is shown in Fig. 9.

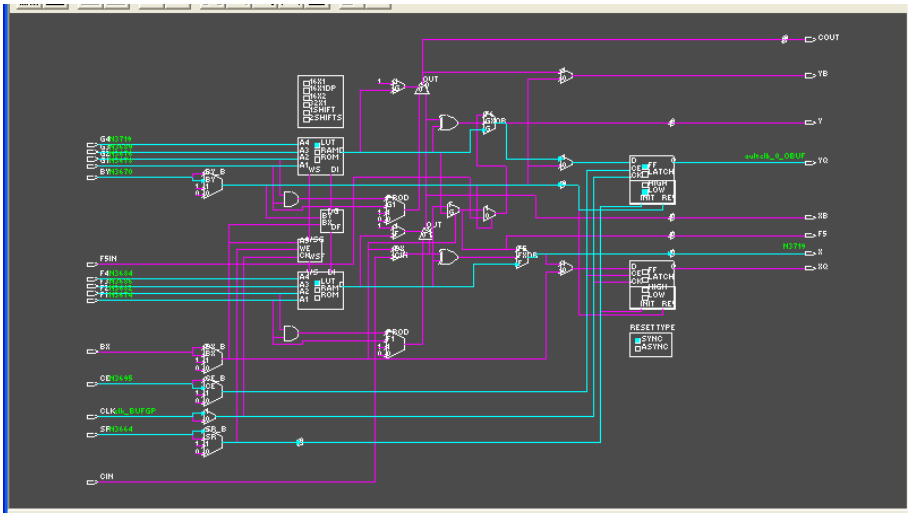

Fig. 9: Logical Placement of CLB unit

The pin layout for the targeted design is shown in Fig. 10. This implementation has dedicated lines of 12 IO lines with V_{cc} and ground pins as seen in Fig. 10. The blue encircled are the allocated line here, wherein the dark gray indicate the dedicated global lines.

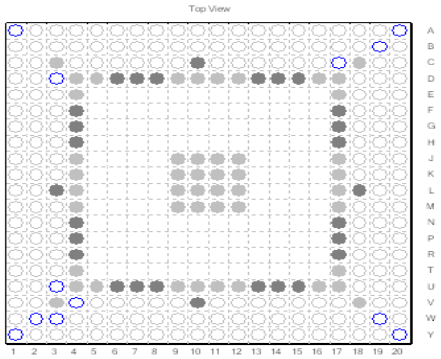

Fig. 10: Pin layout of the implemented design for the targeted FPGA

C) Analysis of developed approach

In the validation of operation performance power measure is critical. The power consumed in a processor unit is defined by,

$$P = \sum_k C_i V_i^2 f_i \quad (1)$$

Where C_i is the capacitance, V_i is the voltage, and f_i is the operating frequency for a set of instruction executed. Here the power is defined as a function of device parameter and the operating frequency of the processing unit. Here, more the operation frequency is more the unit is enabling giving more dissipation of power. However, for reduced computations the operational iteration are reduced which leads to less number of operational cycles and hence reducing the power consumption. The analysis of the power utilization is presented in table 1 below.

Table 1: Observation for power utilization

Instruction density	Power (mW)	
	VA-MPSoC [19]	Vr-MPSoC
4	131.5	114.2
5	274.5	182.8
7	321.4	281.3
9	388.7	311.4
12	451.6	411.3

The comparison of power utilization for different instruction density is shown in Fig. 11

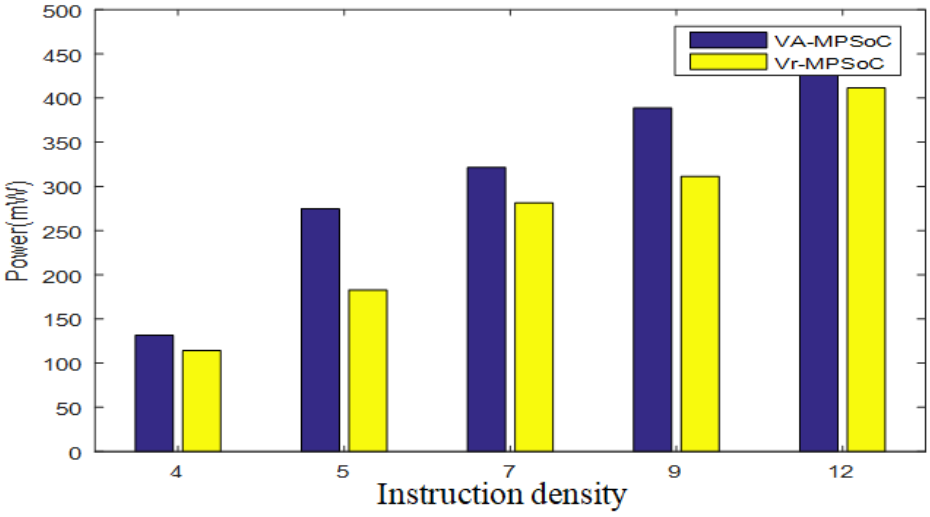

Fig. 11: Comparison of power utilization for different instruction density

The power consumption is proportionally high for large instruction set selection; however the consumption is comparatively less in Vr-MPSoC design due to low delay computation cycles.

The number of computation cycles used in a process is called the latency. The observed latency of the developed approach is summarized in table 2 below.

Table 2: Latency observation for the developed approach

Instruction density	Latency (Cycles)	
	VA-MPSoC [19]	Vr-MPSoC
4	136	112
5	139	106
7	172	119
9	210	178
12	245	221

The comparison of latency for different instruction density is shown in Fig. 12

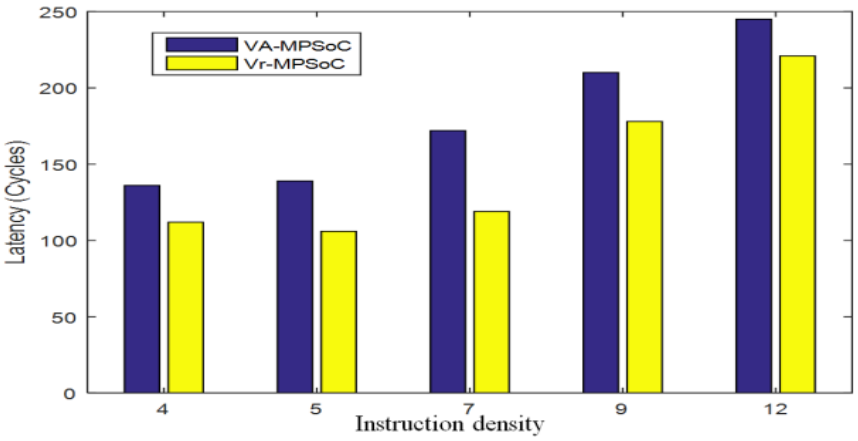

Fig. 12: Comparison of latency for different instruction density

The system performance for a device design is validated by the efficiency of number of processing block per cycle which is termed as throughput. The throughput of a digital system is defined by,

$$THR = \frac{Fmax \times Bsize}{LAT} \quad (2)$$

Where $Fmax$, $Bsize$ and LAT are the maximum operating frequency, block size and latency measured.

Table 3: Throughput observation

Instruction density	Throughput	
	VA-MPSoC [19]	Vr-MPSoC
4	196.4	212.3
5	226.1	272.7
7	270.2	321.1
9	321.4	342.4
12	387.1	396.2

The comparison of the processing throughput for different instruction density is shown in Fig. 13

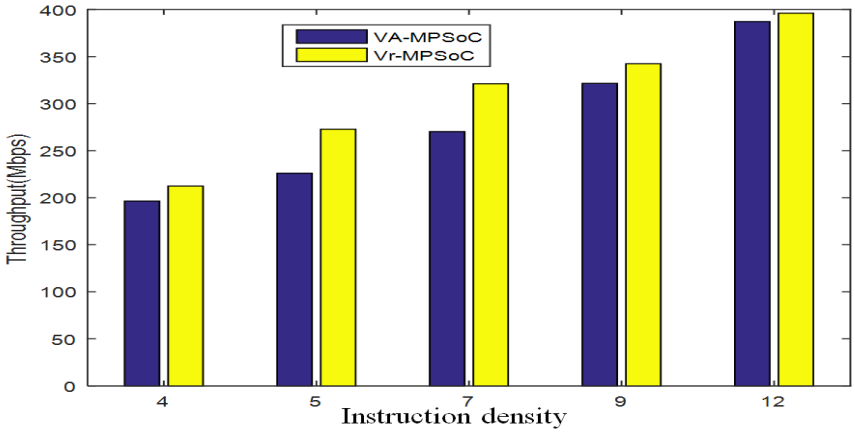

Fig. 13: Comparison of the processing throughput for different instruction density

V. CONCLUSION

This paper presented a new approach of instruction mapping for multi processor system on chip (MPSoC) architecture. The distributed computing has an advantage of faster operation, however the delay parameter in resource allocation constraint its operational performance. To achieve an optimal operational performance in distributing processor units, a new virtualization of clock delay allocation using library mapping in MPSoC design is presented. Stated approach has a significant enhancement in latency minimization and hence reduces the power consumption. This performance illustrates an increase in the processing throughput of the system.

REFERENCES

1. L.Benini and G.De Micheli, "Networks on chip: a new SoC paradigm". IEEE Computer, 35(1):70-78, January 2002.
2. T.Ono, M.Greenstreet, "A Modular Synchronizing FIFO for NOCs", Proceedings of International Symposium on Networks-on-Chip (NOCS), 2009
3. T.Chelcea, S.M.Nowick, "Robust Interfaces for Mixed-Timing Systems", IEEE Transactions on Very Large Scale Integration Systems, 12(8): 857-873, 2004.

4. D.Ludovici, A.Strano, G.N.Gaydadjiev, L.Benini, D.Bertozzi, "Design Space Exploration of a Mesochronous Link for Cost-Effective and Flexible GALS NOCs", Proceedings of Design, Automation and Test in Europe (DATE'10), pp. 679–684, Dresden, Germany, 2010.
5. C.Cummings, P.Alfke, "Simulation and Synthesis Techniques for Asynchronous FIFO Design with Asynchronous Pointer Comparison", SNUG-2002, San Jos`e, CA, 2002.
6. A.Edmanand, C.Svensson, "Timing Closure through Globally Synchronous, Timing Portioned Design Methodology", Proceedings of Design and Automation Conference (DAC), pp.71–74, 2004.
7. P.Caput, C.Svensson, "An On-Chip Delay- and Skew-Insensitive Multicycle Communication Scheme", IEEE Solid-State Circuits Conference (ISSCC), pp.1765–1774, 2006.
8. I.M.Panades, A.Greiner, "Bi-Synchronous FIFO for Synchronous Circuit Communication Well Suited for Network-on-Chip in GALS Architectures", Proceedings of International Symposium on Networks-on-Chip (NOCS), pp.83–94, 2007.
9. D.Ludovici, A.Strano, D.Bertozzi "Architecture Design Principles for the Integration of Synchronization Interfaces into Network-on-Chip Switches", Proceedings of 2nd. International Workshop on Network on Chip Architecture (NoCArc), pp.31–36, New York City, NY, 2009.
10. D.Ludovici, D.Bertozzi, L.Benini and G.N.Gaydadjiev, "Capturing Topology-Level Implications of Link Synthesis Techniques for Nanoscale Networks-on-Chip", Proceedings of the 19th ACM/IEEE Great Lakes Symposium on VLSI (GLSVLSI), pp.125-128, 2009.
11. I.Loi, F.Angiolini, L.Benini, "Developing Mesochronous Synchronizers to Enable 3D NoCs", Proceedings of International Conference on VLSI Design, 2007.
12. D.Ludovici, A.Strano, D.Bertozzi, L.Benini, G.N.Gaydadjiev, "Comparing Tightly and Loosely Coupled Mesochronous Synchronizers in a NoC Switch Architecture", Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip, pp.244-249, 2009.
13. S.Stergiou, F.Angiolini, S.Carta, L.Raffo, D.Bertozzi, G.De Micheli, "XPipes Lite: a Synthesis Oriented Design Library for Networks on Chips", Proceedings of Design, Automation and Test in Europe (DATE'05), pp.1188–1193, 2005.
14. F.Angiolini, L.Benini, P.Meloni, L.Raffo, S.Carta, "Contrasting a NoC and a Traditional Interconnect Fabric with Layout Awareness", Proceedings of Design, Automation and Test in Europe (DATE'06), March 2006.
15. "Specification of optimized GALS interfaces and application scenarios", GALAXY Project deliverable D3, online at http://www.galaxyproject. org/publ deliv.html
16. J.Ebergen, "Squaring the FIFO in GasP", Proceedings of International Symposium on Asynchronous Circuits and Systems, pp.194–205, 2001.
17. C.E.Molnar, I.W.Jones, W.S.Coates, J.K.Lexau, "A FIFO ring performance experiment", Proceedings of International Symposium on Asynchronous Circuits and Systems, pp.279–289, 1997.
18. R.Apperson, Z.Yu, M.Meeuwsen, T.Mohsenin, B.Baas, "A scalable dual-clock FIFO for data transfers between arbitrary and haltable clock domains", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 15(10), pp.1125–1134, 2007
19. Johannes Ax, Gregor Sievers, Julian Daberkow, Martin Flasskamp, Marten Vohrmann, Thorsten Jungeblut, Wayne Kelly, Mario Porrmann and Ulrich Ruckert, "CoreVA-MPSoC: A Many-core Architecture with Tightly Coupled Shared and Local Data Memories", IEEE Transactions on Parallel and Distributed Systems, Post-Print , December 2017.

AUTHOR PROFILE

Sukanya. K, presently working as Associate professor in the department of Electronics and Communication Engineering at TKR College of Engineering & Technology, Medbowli, Meerpet, SaroorNagar, Hyderabad, Telangana State, INDIA. She has 7 years of teaching experience. She is associated with ISTE as life member. She has obtained B. Tech. degree in Electronics and Communication Engineering from Jayamukhi Institute of Technological Sciences, Warangal, Jawaharlal Nehru Technological University Hyderabad, in 2006, M.Tech. degree in Embedded Systems from Ramappa Engineering College, Warangal, Jawaharlal Nehru Technological University Hyderabad, in 2011 and my area of Research interest is Embedded Systems, Ph.D (ECE) from Jawaharlal Nehru Technological University, Hyderabad and it is my part of Research work.

Dr. G. Laxminarayana, presently working as Principal of Anurag College of Engineering. He has obtained BE from Osmania university, M.Tech from Indian Institute of Science, Bangalore and Ph.D from JNTUH under the guidance of Dr. K. Lalkishore (VC of JNTUA). He has 5 years of industrial experience and 30 years of teaching experience. He worked in Osmania University from 1979 to 1998. He worked as Head of the department, ECE at Sreenidhi Engineering College, VBIT and Aurora. He also worked as Director of Aurora Scientific Technological and Research Academy and Principal of Holy Mary Institute of Technology. He is an industrial consultant in instrumentation and worked in South Central Railways. He is associated with ***IETE for last 20 years*** and also member of IEEE, ISOI and ISTE. Presently he is an ***Executive Committee member*** in the present body of Hyderabad IETE chapter and ***R&D subcommittee chair at IETE Hyderabad***. He is supervising 12 Ph.D students and published various papers in International journals and reputated National journals.