Design of Multicore Processor using Multithreading Technique

Manoj Kumar Gouda, D. Yugandhar

Abstract : The multicore design has become a design philosophy in engineering applications. Speedup has been achieved by increasing clock speeds and, more recently, adding multiple processing cores to the same chip. Although chip speed has increased exponentially over the years, that time is ending and manufacturers have shifted toward multicore processing. However, by increasing the number of cores on a single chip challenges arise with memory and cache coherence as well as communication between the cores. A multi-core processor is a processing system composed of two or more independent cores. One can describe it as an integrated circuit to which two or more individual processors (called cores in this sense) have been attached. This paper discuss the advantage in the transition from non pipelined processor to single core pipelined processor, and the transition from single core pipelined processor to multicore pipelined processor and finally ends in with designing a quadcore processor. It begins with the discussion of implementation of a non pipelined processor. Secondly we discuss the process of converting it into a pipelined processor and the shared memory issues are discussed. Finally provides the design details of all the phases of a multicore processor with quad port memory design, including performance achievement achieved by this transition. The design is done on xilinx Spartan xc6slx45-csg324-4 FPGA and it performance characteristics are anlaysed . The designed Quad core performance issues like area, speed and power dissipation are also presented.

Keywords - Mulit-core processor, Pipelining, Quadcore,FPGA.

I. INTRODUCTION

The exponentially growing number of transistors on chip predicted by Moore's law has opened up the possibilities for implementing increasingly complex systems on chip. These systems require high performance levels and low power consumption. In multiprocessor systems, each processor functions at a moderate speed and the overall throughput of the system is increased by the multiplicity of processors. Even though a single processor runs at higher clock speeds, the cost of designing a multiprocessor system-on-chip remains smaller than that of designing a single processor. The multiplicity of processors as well as the importance of reconfigurability in error correction and system enhancement has made the design and implementation of a multiprocessor system on a Field Programmable Gate Array (FPGA) a hot research topic. The specification of a multiprocessor system needs to be described at the lowest level in order to be downloaded onto the FPGA. Commercial state-of-the-art tools take as input a higher level description of the system, the Register Transfer Level (RTL), and convert it to a code that can be downloaded onto the FPGA.

Manuscript Received on October 2014.

Manoj Kumar Gouda, M.Tech (VLSI Sys Design), Department of ECE, AITAM, Tekkali, AP, India.

D. Yugandhar, Department of ECE, Assoc. Prof., Tekkali, AP, India.

Nevertheless, the increasing complexity of today's applications and systems makes creating an RTL description of these computationally demanding systems inadequate. Therefore, there is a need to move to a higher level of specification. However, this creates a gap between the traditionally used RTL level and the higher level of implementation, the system level. The main purpose of the project proposed in the fall term was to put forward a tool that will automate the transition between the system level and the RTL level. This tool would enable designers to build complex multi-processor systems on a higher level saving time and effort. In computing, a processor is the unit that reads and executes program instructions, which are fixed-length (typically 32 or 64 bit) or variable-length chunks of data. The data in the instruction tells the processor what to do. The instructions are very basic things like reading data from memory or sending data to the user display, but they are processed so rapidly that we experience the results as the smooth operation of a program. Processors were originally developed with only one core. The core is the part of the processor that actually performs the reading and executing of the instruction. Single-core processors can only process one instruction at a time. (To improve efficiency, processors commonly utilize pipelines internally, which allow several instructions to be processed together, however they are still consumed into the pipeline one at a time.) A multi-core processor is a processing system composed of two or more independent cores. One can describe it as an integrated circuit to which two or more individual processors (called cores in this sense) have been attached.[1] The cores are typically integrated onto a single integrated circuit die (known as a chip multiprocessor or CMP), or they may be integrated onto multiple dies in a single chip package. A many-core processor is one in which the number of cores is large enough that traditional multi-processor techniques are no longer efficient — this threshold is somewhere in the range of several tens of cores — and probably requires a network on chip.

II. SINGLE CORE PIPELINED ARCHITECTURE

Single core processor contains four stages as follows

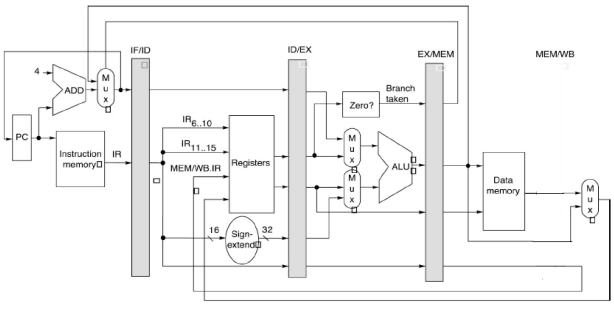

Fig. 1: Four Stage Pipelined MIPS Processor

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

2.1 Instruction Fetch

Send out the PC and fetch the instruction from memory into the instruction register (IR); increment the PC by 2 to address the next sequential instruction. The IR is used to hold the instruction that will be needed on subsequent clock cycles; likewise the register NPC is used to hold the next sequential PC.

2.2 Instruction Decode

Decode the instruction and access the register file to read the registers. The outputs of the general purpose registers are read into two temporary registers (A and B) for use in later clock cycles.

2.3 Instruction Execute

2.3.1 Arithmetic Logic Unit

The ALU performs the actual numerical and logic operation such as 'add', 'subtract', 'AND', 'OR', etc. Uses data from A and B Register to perform arithmetic and logical operations and stores result of operation in C Register. Carry Skip Adder is used for addition and subtraction, Modified Booth Multiplier is used for Multiplication. and barrel shifter is used for rotate operations.

2.3.1.1 Advantages

1. Less Area/Power.
2. Adder of n stages divided into stages of selected lengths with Simple Ripple Carry.
3. Each group generates "Group Carry-Propagate"=1 if all pi=1 in Each Group.
4. This signal allows Incoming Carry to "Skip" Stages within group and generate carry out

2.3.2 Control Unit

The control unit is one of the most important parts of a microprocessor for the reason that it is in charge of the entire process, that is the machine cycle. The cpu deals with each instruction it is given in a series of steps. Each step is repeated for each instruction. This series of steps is called the machine cycle. It involves:

1. fetching an instruction from memory;
2. decoding the instruction;
3. transferring the data;
4. executing the instruction. The control unit makes sure that all of those actions are carried out. It also manages all the other components on the cpu.

III. MULTICORE ARCHITECTURE

3.1 BASIC CONCEPTS

A multi-core processor is an integrated circuit (IC) to which two or more processors have been attached for enhanced performance, reduced power consumption, and more efficient simultaneous processing of multiple tasks (see parallel processing). A dual core set-up is somewhat comparable to having multiple, separate processors installed in the same computer, but because the two processors are actually plugged into the same socket, the connection between them is faster. Ideally, a dual core processor is nearly twice as powerful as a single core processor. In practice, performance gains are said to be about fifty percent: a dual core processor is likely to be about one-and-a-half times as powerful as a single core processor.

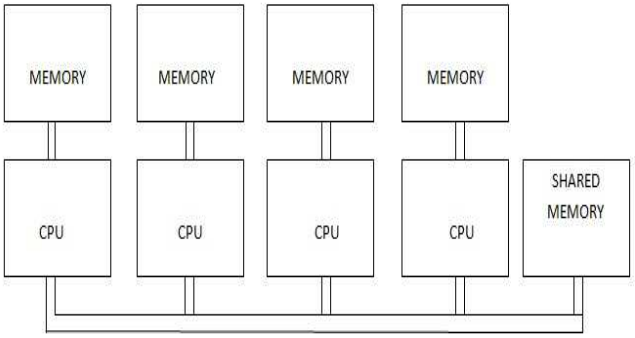

Fig. 2: Simple Quadcore Processor with Shared Memory

In this figure 2 the orientation of the cores in a multi-core processor is visualized. On the right side the CPU cores are seen and on the left the Level 2 Caches can be seen. Both Intel and AMD have been slowing down the rate for increasing the clock speeds of processor. Part of this is due to the limitations of the current technology and designs. To try and keep up to pace with future developments, both companies are introducing dual-core processors in 2005. For some time now, the benefits of multiple processors have been seen in the server environment. By having multiple processors on a single server, the tasks running on the server can be divided between the processors to allow the system has a whole to function faster. The goal of a dual-core CPU is to take two physical processors and integrate them on one physical chip.

3.2 THREADING

Before going into the benefits and drawbacks of dual-core or multiple processors, it is important to understand the concept of threading. A thread is simply a single stream of data through the processor on the system. Each application generates its own or multiple threads depending upon how it is running. With current multitasking, a processor can only handle a single thread at a time, so the system rapidly switches between the threads to process the data in a seemingly concurrent manner. The benefit of having multiple processors is that the system can handle more than one thread. Each processor can handle a separate stream of data. This greatly increases the performance of a system that is running concurrent applications such as a server. Intel implemented something they called Hyper threading[7]. This is not the same as multithreading. Instead it is a technology embedded within a single core processor to make it appear to the system as if it had multiple processors. What this really did was speed up the rate at which the system could switch between multiple threads thus boosting multitasking on personal computers.

3.3 IMPLEMENTATION OF MULTIPORT SHARED MEMORY

Figure 3 shows the chip area utilization for a four-processor system with shared memory. The multicore system is designed, simulated and implemented on Xilinx FPGA – chips of Spartan 6 family by means of Web PACK design environment, simulator ISim and Verilog source code. The multiport shared memory is implemented by means of block Select RAM cells of the FPGA –chip. Each block Select RAM cell in FPGA is a fully synchronous memory, as illustrated in Fig.3. The two ports have independent inputs and outputs and are independently clocked.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

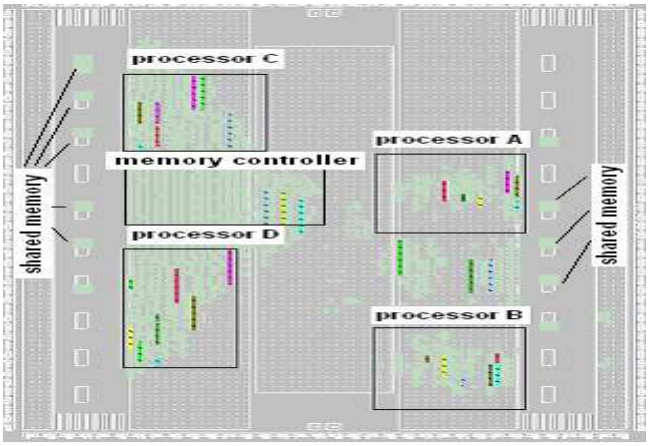

Fig. 3: Simple Quadcore Processor with Shared Memory

3.3.1 DUAL-PORT MEMORY

The dual-port memory configuration consists of two data access ports (two read/write ports) sharing a single clock domain (wr_clk). The write address bus from each data access port (a_wadr and b_wadr) is used for both read and writes operations. The read enable (a_rdblk, a_rdb, b_rdblk, and b_rdb) and write enable (a_wrblk, a_wrb, b_wrblk, and b_wrb) signals are used to select between either read or write operation for each data access port. Figure 4 shows the corresponding ports of a dual-port memory block.

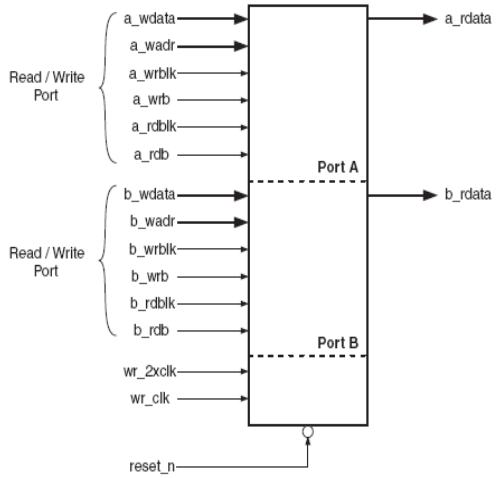

Fig. 4: Dual-Port Memory Block Interface Signals

3.3.2 QUADPORT MEMORY

The quad-port memory[1][2] configuration consists of two data access ports, each with a separate write port and read port,clocked by separate write (wr_clk) and read (rd_clk) clocks. For each data access port, there are separate address busses used to perform read (a_radr and b_radr) and write (a_wadr and b_wadr) operations. The read enable (a_rdblk, a_rdb, b_rdblk, and b_rdb) and write enable (a_wrblk, a_wrb, b_wrblk, and b_wrb) signals are used to activate the read and write operations for each data access port. Figure 5 shows the corresponding ports of a quad-port memory block.

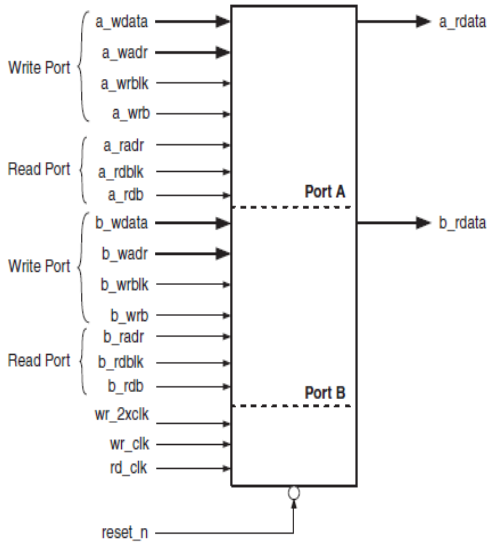

Fig. 5: Quad-Port Memory Block Interface Signals

Signal	Bits	In/Out	Description
a_wdata	variable	IN	Write data bus
a_wadr	8	IN	Write / dual-port memory address bus
a_wren	1	IN	Active high data enable
a_rdata	variable	OUT	Output data bus
a_radr	8	IN	Output address bus (quad-port memory mode only)
a_rden	1	IN	Output register enable A
b_wdata	variable	IN	Write data bus
b_wadr	8	IN	Write / dual-port memory address bus
b_wren	1	IN	Active high data enable
b_rdata	variable	OUT	Output data bus
b_radr	8	IN	Output address bus (quad-port memory mode only)
b_rden	1	IN	Output register enable B
wr_2xclk	1	IN	2x write clock
wr_clk	1	IN	Write port data clock / multiplexer select
rd_2xclk	1	IN	2x read clock (quad-port memory mode only)
rd_clk	1	IN	Read data clock (quad-port memory mode only)
reset_n	1	IN	Reset signal (active low)

Fig. 6: Memory Interface Signals

Since implementation of multi-port memories relies on the the embedded memory block being clocked at twice the data clock rate, a double frequency clock needs to be generated. The original data clock input (wr_clk) is easily doubled in frequency to generate the required wr_2xclk signal

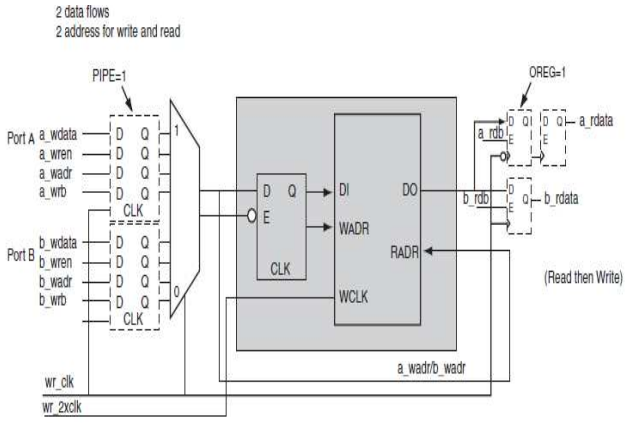

Fig. 7: Block Diagram of Memory

3.3.3 USING A SHARED MEMORY

This consists of a global address space which is accessible by all N processors. A processor can communicate to another by writing into the global memory where the second processor can read it.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

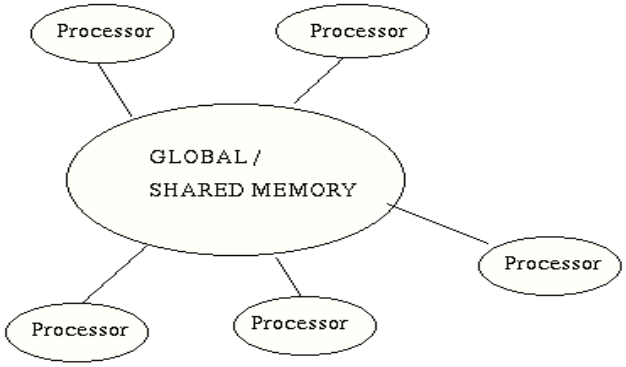

Fig. 8: Multicore Architecture

Shared memory [5]solves the interprocessor communication problem but introduces the problem of simultaneous accessing of the same location in the memory. Consider.

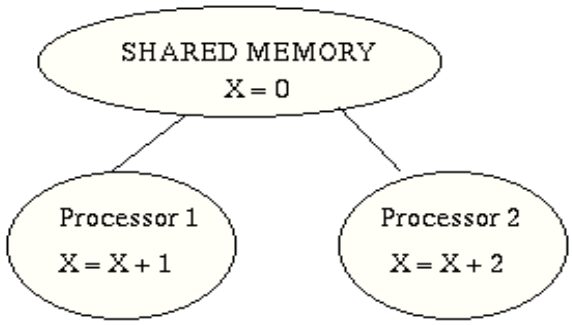

Fig. 9: Example of Shared Memory

i.e. x is a shared variable accessible by P1 and P2. Depending on certain factors, x=1 or x=2 or x=3.

1. if P1 executes and completes x=x+1 before P2 reads the value of x from memory then x=3 similarly if P2 executes and completes x=x+2 before P1 reads the value of x from memory then x=3 .
2. if P1 and P2 read the value of x before either has updated it then the processor which finishes last will determine the value of x.
   if P1 finishes last the value is x=1
   if P2 finishes last the value is x=2

In a multiuser, real time environment the processor which finishes last would vary from run to run - so the final value would vary. Also, even if they finish at the same time only one value of x can be stored in the location for x. This gives rise to Non-Determinacy. When a parallel program with the same input data yields different results on different runs. Non-determinacy is caused by race conditions.A race is when two statements in concurrent tasks access the same memory location, at least one of which is a write, and there is no guaranteed execution ordering between accesses. The problem of nondeterminancy would be solved by synchronising the use of shared data. That is; if x=x+1 and x=x+2 were mutually exclusive statements i.e. could not be executed at the same time, then x=3 always.

3.3.4 Solution to Shared Memory Problem

Here each processor has access to any variable residing in the shared memory. So if processor x wishes to pass a number to processor y it must be done in two steps. Processor x writes the number into the shared memory at a given location accessible to processor y, which then reads the number from that location. During the execution of a parallel algorithm, the N processors can access shared memory for reading / writing data and / or results. All processors can gain access to the shared memory simultaneously if the memory locations they are trying to read from or write into are different. However we can get problems when two or more processors require access to the same memory location simultaneously.

Example:

Let x be a variable that can be accessed by two processors P1 and P2

sNow consider the assignment x := x + 1, which is normally done in three stages.

1. copy value of x into some register
2. add 1 to value on register
3. store value on register at the address for x

Say now that P1 and P2 both execute such an assignment, assume x=0 initially

What is the final result?

P1 copies value of x (= 0) into its register
P2 copies value of x (= 0) into its register
P1 adds 1 to its register --- these two at the same time
P1 stores value of x (= 1)
P2 adds 1 to its register --- these two at the same time
P2 stores value of x (= 1)

Giving a result of 1 rather than 2. This is because P2 reads the value of x (=0) before P1 has updated it.Therefore depending on whether 2 or more processors can gain access to the same memory location simultaneously, we have 4 subclasses of shared memory computers

IV. SYNTHESIS AND SIMULATION

We have designed a FOUR core processor. We added 5 numbers using 4 cores. It increases the speed of getting result. The simulations in Isim are shown in figure below. We added 1st two numbers in 1st core ,next two numbers in second core, the results are passed to 3rd core and remaining number in 4th core. A maximum of 5% of the total LUT's are utilized and 30 out of 218 IOB's are utilized. Overall power utilisation is 80mW and efficiency of a multicore processor is 3 times compared to a single core processor. To determine complexity and scalability, we examine the FPGA mapping of the design. In the first step, we start by choosing a particular architectural technique that provides a certain subset of the required functionalities. The second step considers the structural design choices of each architectural technique in terms of the temporal and spatial parallelism necessary to meet throughput and latency requirements. In the third step, we take into consideration the logic-implementation-specific choices driven by the constraints of the target FPGA chip.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.

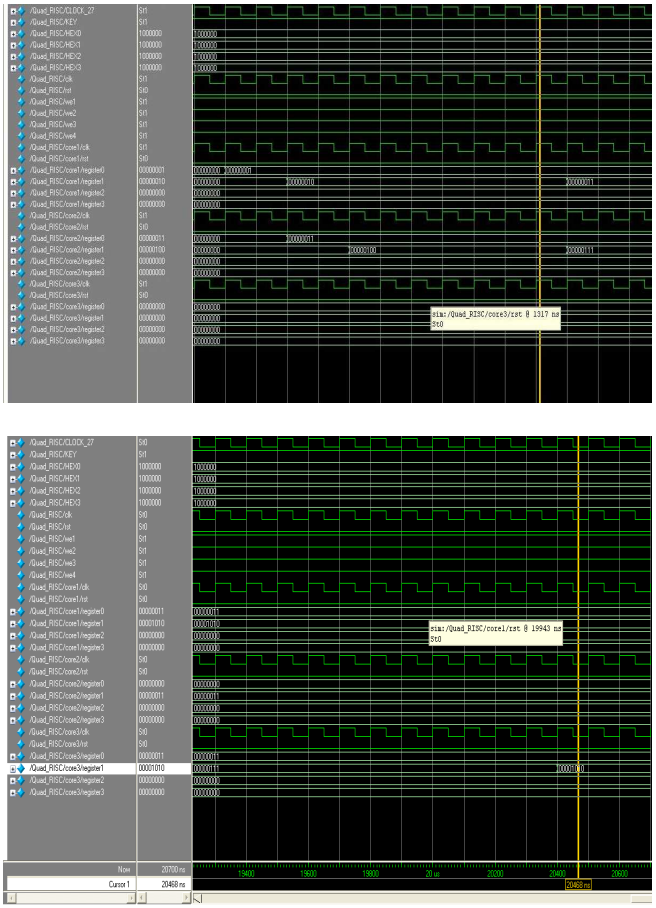

Fig. 11: Simulation Output of Addition

DEVICE UTIZATION SUMMARY

Slice Logic Utilization	Used	Available	Utilization
Number of Slice Registers	2,913	54,576	5%
Number of Slice LUTs	16,502	27,288	60%
Number used as logic	16,240	27,288	59%
Number of bonded IOBs	30	218	13%

Timing Report:

Minimum period: 5.663ns (Maximum Frequency: 176.577MHz)

Minimum input arrival time before clock: 5.493ns

Maximum output required time after clock: 5.179ns

V. CONCLUSION

The designed quad processor core is very efficient and powerful core architecture, which is analyzed by using Xilinx software and simulation by using Isim simulator. Although the quad processor core was based on well-proven superscalar architectural techniques, including parallel execution and register renaming, these techniques were originally devised for fine-grained parallelism among instructions. The designed quad core implementation using FPGA which is very advantageous to the architecture ,because it uses the parallel processing and pipielinig techniques are easily implemented in FPGA.The concurrent operation of quad processor gives the high speed and takes less power.

REFERENCES

[1] Tai-Hua, Lu, Chung-Ho Chen, KuenJong Lee. “Effective Hybrid Test Program Development for Software-Based Self-Testing of Quad Cores” ,IEEE Manuscript received April 03, 2012, revised August 14, 2012, first published December 18, 2012.

[2] Gohringer, D., Hubner, M.Perschke, T., Becker. J. “New Dimensions for Quad core Architectures Demand Heterogeneity”, Infrastructure and Performance through reconfigurability The EMPSoC Approach”. In Proc of FPL 2010, PP.495-498, Sept 2010.

[3] Lysaght, P. Blodget, B. Mason, J.Young, B.Bridgford. “Invited Paper: Quad core design Methodologies and CAD Tools for Dynamic Reconfiguration of Xilinx FPGAs”. In Proceedings of FPL 2009, August 2009.

[4] D. Tullsen, S. Eggers, and H. Levy, “Simultaneous Multithreading: Maximizing On- Chip Parallelism,” Proc. 22nd Ann. Int’l Symp. Computer Architecture, ACM Press, New York,1995, pp. 392-403.

[5] J. lo, S. Eggers, J. Emer, H. Levy, R. Sstamm, and D. Tullsen.”Converting thread level parallelism into instruction-level parallelism via simultaneous multithreading”. ACM Transactions on Computer Systems, 15(2), pp. 323-354, August 1997.

[6] Lance, Hammond, Basem, Ku.umen “ANayfeh, Kunle Olukotun.A Single-Chip multiprocessor. IEEE Computer”, vol. 30, no. 9, pp. 79--85, September1997.

[7] J. Borkenhagen, R. Eickemeyer, and R. Kalla : “A Multithreaded PowerPC Processor for Commercial Servers, IBM Journal of Research and Development”, November 2000, Vol. 44, No. 6, pp.1995.

Manoj Kumar Gouda, pursuing his M.Tech (ECE) in Department of ECE at Aditya institute of Technology and Management, Srikakulam, A.P., India. His research interest includes VLSI System Design, signal processing.

Dasari Yugandhar, working as Assoc. Professor, Department of ECE at Aditya institute of Technology and Management, Srikakulam, A.P., India. He is presently pursuing Ph.D from Berhampur University. His research interest includes, signal processing, speech processing and VLSI Design.

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication Pvt. Ltd.