ISSN NO. 2320-5407

Journal homepage: http://www.journalijar.com
Journal DOI: 10.21474/IJAR01

INTERNATIONAL JOURNAL OF ADVANCED RESEARCH

RESEARCH ARTICLE

LITERATURE SURVEY OF MICROPROCESSOR USING FPGA WITH PIPELINING

***Tanushree Girkar[1], and Sneha Nagar[2].**

1. Department of Electronics & communication Oriental University, Indore(M.P.)
2. Asst prof, Department of Electronics & communication, Oriental university, Indore. (M.P.)

Manuscript Info

Manuscript History:

Received: 25 April 2016
Final Accepted: 19 May 2016
Published Online: June 2016

Key words:
FPGA,VHDL,Xilinx

Corresponding Author

Tanushree Girkar.

Abstract

This paper covers motivation for VLSI and FPGA both. We can improve the speed of processor by the help of proposed design using the technique of pipelining in 5 stages .This is done to improve the overall speed of the processor. As the speed increases less amount of time is required for processing hence we get a fast processor. We have also mentioned about the languages used for the design. Every module is designed in VHDL.Xillinx is the main software used for function.

Introduction:-

VLSI design engineers have to deal with three things i.e. speed, area and power. If one of the factors is increased other may affect. So there should be a balance between these three. There are reasons that Fpga becomes so popular since there can be parallel programming in Fpga which can speed more. The program is not burned in Fpga rather it is downloaded. There are configurable logic blocks and interconnections in Fpga to make our task easy. The smallest abstraction level in Fpga is logic gates. Hence, to meet the market requirements Fpga is best because it is economic only one time cost required for the hardware. As designs increase in complexity, the difficulty of describing what the design does, and demonstrating the design does also increases. The main factor for preferring VLSI design is a single chip solution, which is supporting to create our own processor. The thing associated in this paper is pipelining, which will improve the speed of the processor. There are segments in pipelining where each segment executes its own operation concurrently with other segment. Pipelining can be buffered or unbuffered depends on us which one to be used.

Languages used:-

VHDL is a vast language. It works at RTL level, where the smallest unit is either registers, adders, subtractors.The most important thing is not only there structure but the timing relations of input and output. There are some requirements with hardware description language then only we can appreciate the language. The four requirements are abstraction, modularity, concurrency and hierarchy.

VHDL is used to write text models that describe a logic circuit. There is a simulation program for test the logic design using simulation models to represent the logic circuits that interface to the design. This collection of simulation models is commonly called a test bench. VHDL has filed input and output capabilities, and can be used as a general-purpose language for text processing, but files are more commonly used by a simulation test bench for stimulus or verification data. In this case, it might be possible to use VHDL to write a test bench to verify the functionality of the design using files on the host computer to define stimuli, to interact with the user, and to compare results with those expected. Many designers leave this job to the simulator. VHDL is a Dataflow language which all runs sequentially, one instruction at a time. It is capable of describing very complex behaviour. For designing any digital system VHDL is the best language.

FPGA:-

It stands for Field programmable gate array. It consists of mainly two things. The first one is Integrated circuits and second is configurable blocks which are programmable. Here the field stands for site or lab where a device can be programmed or installed. FPGA is also used by students as it is easy to learn and various hardware can easily understand by its help. The basic unit of Fpga is matrix. Layout of unit is repeated in matrix form. The three major blocks of FPGA are logic module, Input/output ports and Interconnections. We will only program the interconnections. So to make the interconnection first thing is either making or breaking the function and define the function of logic block. The three programming methods of FPGA are SRAM based, Antifuse technology and EEPROM.The former two are used frequently. Today's FPGA now contains approximately 300,000 logic blocks and around 1100 inputs and outputs. The basic architecture of FPGA consists of three major blocks: programmable logic block which implements the logic functions, programmable routing (interconnects) to implement these functions and I/O blocks to make off-chip connections. A flow chart of typical FPGA architecture is shown in figure 1.

SRAM based programming:-

It uses concepts of Pass transistors, Transmission gates and multiplexers. RAM is also known as single shift transistor since FPGA contains many of them and the logic travels through them. The program which we write using any software coverts into form of bit streams which is serially loaded into SRAM cells. The disadvantage of SRAM is that it requires greater area in FPGA and delay is more.

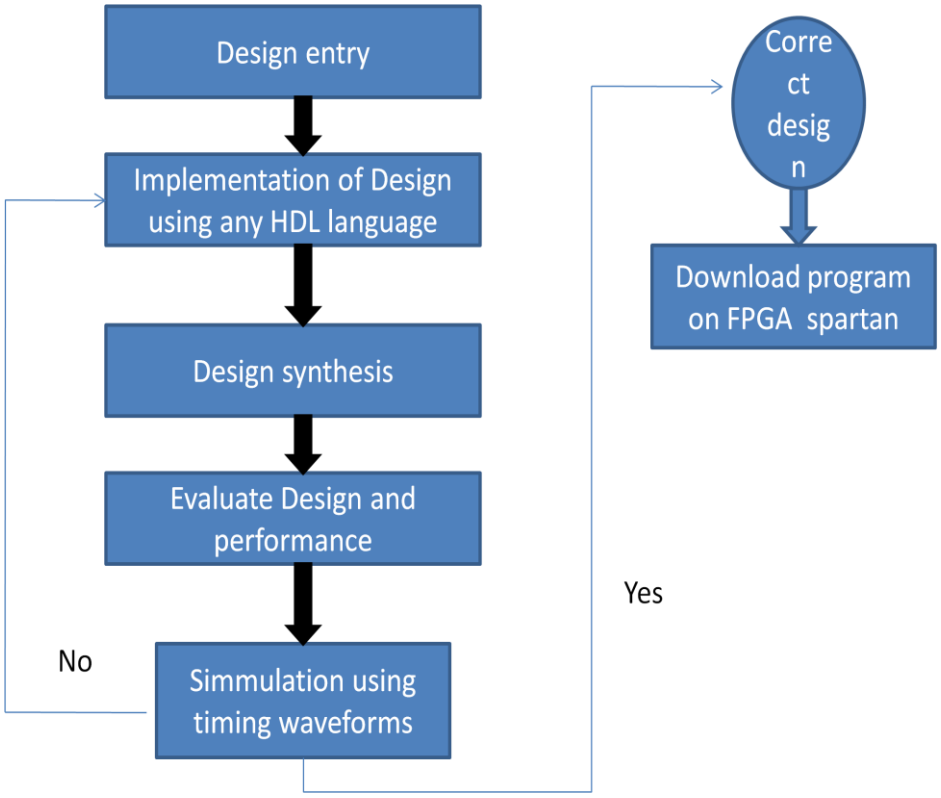

Fig 1:- Flow chart of Fpga.

Pipeling:-

Pipeling is something we come across day to day life. To design a CPU we need instruction set, no of registers and their details,datapath i.e. data will from which source to which destination and the control circuits for execution of instructions there can be two types hardware control signals and software control signals. Distributed computer is mostly used. We want to do more no of tasks in less amount of time for that we require a new concept i.e. pipelining which will help in reducing execution time.

Let's take an e.g. to understand this suppose we have a task t to be completed in time т lets divide the task in four subtasks so that our work will reduces. The subtasks are named as t1, t2, t3 and t4.Suppose we have four computing blocks s1, s2, s3, s4 to complete the task. If s1 is capable for computing the task for t1 similarly s2, s3, s4 will compute for t1, t2, t3 respectively. Hence, the total time needed for task t is 4 т.See another option how we can improve this time.

If the tasks are organized in a manner than it has to go through the pipeline. Now, whenever the tasks will come the subtask t1 is executed and then t2 is and so on. The time when more no of tasks are waiting for completion we can do a little thing is that If t1 of t is completed the s1 is free to perform t1 of any new task. Pipeline is full when every computing task is filled with some work. At a time interval of т we will get the output and in case of no pipelined architecture we get the output after every 4 т.If I have k stages, every stage takes т,if n no of jobs are there then time taken will be n т in case of non-pipelined architecture but in case of pipelined architecture we have total time n+(k-1) т.Thus,we get tremendous gain in case of a pipelined architecture. The performance parameters of pipeline are speed up, throughput and efficiency. Speed-up is indirectly proportional to time taken. Speed-up is defined as time taken for a given computational non pipelined functional unit divided by time taken for same computation pipeline. It is an indicator of how efficiently the resources of the pipeline are used. If a stage is available during a clock period, then its availability becomes the unit of resource.

$$\text{Efficiency} = \frac{\text{nk}}{k[\text{k} + \text{n} - 1]} = \frac{n}{k + n - 1}$$

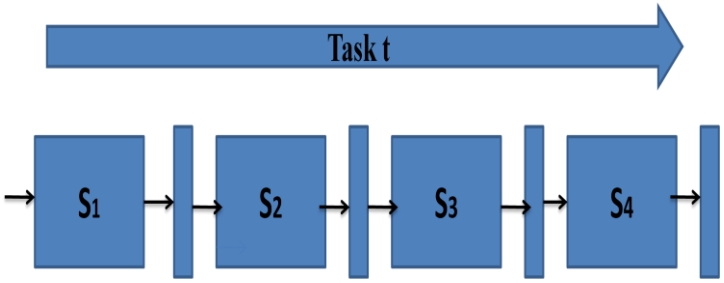

Fig 2:- Pipeling example.

Efficiency is minimum when n = 1.Throughput is the average number of results computed per unit time. For n inputs, a k-staged pipeline takes [k+(n-1)]T time unitsThen,Throughput = n / [k+n-1] T = nf / [k+n-1] where f is the clock frequency .Instruction pipelining is the special case of pipelining. The execution of a stream of instructions can be pipelined by overlapping the execution of current instruction with the fetch, decode and operand fetch of the subsequent instructions. It is also called instruction look-ahead. One of the examples is 8086 processor. Processor pipelining refers to the processing of same data stream by a cascade of processors each of which processes a specific task. The data stream passes the first processor with results stored in a memory which is also accessible by the second processor. The second processor then passes the results to the third and so on.

RISC v/s CISC Architecture:-

RISC stands for reduced instruction set computation. The IBM was the first company to design RISC processors in 1970's.Afterwards researches have been made in universities of Berkeley and Stanford to give basic architectural models. It has very simple instructions which are friendly for the programmer, whereas CISC stands for complex instruction set which give emphasis on power features. It has less no of instructions but they are complex to understand.RISC is nowadays used by many companies it also provides uniformity of the instruction where same type of instruction are used. It has simple set of operations; one single task is performed at one time. Register based architecture with adding instruction. The examples of sum SPARC, HPPA-RISC, Motorola Power PC, CDC.The examples of Motorola 680*0 series of processors, Intel 8086and vax to minimize code size. It makes assembly language easy. Motorola works on semiconductor technology. Some complex instructions are slower than set of simpler instructions to perform the equivalent task. RISC processors have shorter design cycles.RISC instructions take less clock cycles than CISC instructions. CISC instructions take up to 3 to 12 times longer. CISC processors provide a

variety of addressing modes, which leads to variable-length instructions. In contrast to the RISC processors, CISC processors allow constants as well as operands that are either in memory or in registers. The instruction size depends on where the operands are, or whether it is a constant. For example, instruction length increases if an operand is in memory as opposed to in a register. This is because we have to specify the memory address as part of instruction encoding, which takes many more bits. Similarly, it the operand is a constant, it is stored as part of the instruction, which again increases the length of the instruction depending on the size of the constant. The size of the instruction will depend on the number of addresses and whether these addresses identify registers or memory locations. Since RISC processors use register-to-register operations, we need to have a large number of registers. A large register set can provide more no of opportunities for the compiler to optimize their usage. Another advantage with a large register set is that we can minimize the overhead associated with procedure calls and returns. In case of complex instructions hardware complexity is handled by microprogramming, which can reduce the impact of memory access latency, offers flexibility, multiple members of the same family. Both RISC and CISC try to solve the same problem. One of the main concerns of designers of RISC processors was to have pipeline. Most of the current processors are not typical RISCs or CISCs but try to combine advantages of both approaches.

Conclusion:-

A pipelined architecture is a better option for us to support RISC processors. In this survey we observed the motivation for vlsi design and FPGA. If we compare microcontroller programming to FPGA then FPGA is better because program can be run and erased easily, hence n no of designs can be developed through FPGA. The next step to this survey will be implementation of RISC processor with the help of FPGA. We can also add more no of stages of pipeline to improve the gain.

References:-

1. **David A. Patterson and John L. Hennessy,** Computer Organization & Design ISBN 1-55860-428-6, p 476-501, 525-256.
2. **Rose Hulman**, Computer Organization & Design
3. **M. Jhand, S. Lehner, W. Muller, O. Schlett,** A 32-b RISC/DSP microprocessor with reduced complexity **06 August 2002 .**
4. **B.Ramamurthy**,Basic pipelining,
5. **XILINX datasheet library,** http:// www.xilinx.com/ partinfo/4000.pdf .