A REVIEW ARTICLE 32 BIT RISC PROCESSOR DESIGN

Amit Yadav[1], Deepak Sharma[2]
Lord Krishna College of Technology

Abstract - In this paper, we propose 32-bit pipelined RISC processor using VLIW architectures. This processor is especially used for both D.S.P applications and general purpose applications. Reduced instruction is the main criteria used to develop in this processor. With a single instruction scheme, more executions can be done using S.I.M.E. processor consists of the blocks namely program counter, clock control unit, ALU, IDU and registers. Advantageous architectural modifications have been made in the incrementer circuit used in program counter and carry select adder unit of the ALU in the RISC CPU core. In this paper, we have extended the utility of the processor towards convolution and correlation applications, which are the most important digital signal processing applications.

Keywords - *RISC, VLIW, SIME, Convolution, Correlation.*

I. INTRODUCTION

A processor is something which processes the given task. It is developed to make human task easy, and when it processes micro things it is called as a microprocessor. It contains basically two things. One is the CPU and second is the memory .CPU stands for central processing unit. It controls the whole processor; we can also call it a human brain. The design of processor needs the designing of CPU and memory. The memory can be static or random. Design of CPU can be done in different ways but its fundamental operation remains the same. CPU is designed on small scale and large scale integration. Processors are designed with enhancements like decreasing power consumption, increasing more pipeline stages, increasing speed and minimizing the chip area. The basic operation performed by a processor is fetching, decoding and executing the given instruction. After that the output will store in memory. For storing the results in memory some mechanism is applied.

II. DESIGN RULES IN VLSI

The rules are made to check whether the design is correct or not. After full verification of each rule the design goes to the fabrication lab for fabrication. This process is called as Electronic design Automation. These rules basically check layout not the schematic. There are lambda based rules, Meta rules and micro based rules. Amongst them most popular is lambda based rules. These rules play an important role because they are an interface between design engineers and fabrication engineers. There are design rules checking software. There are basic three rules for width, length and masks. Some of the lambda rules are as follows-

- Well to well spacing should be 2 χ
- Well to poly spacing should be 2 χ
- Well to metal spacing should be 3 χ
- Poly-active minimum spacing should be 1 χ
- Poly overlap spacing should be 2 χ

Pipelining Processors:

[4]The pipelining is a kind of parallel processing used to increase the speed of the processor. It can be either software or hardware. It consists of some stages one after another. With pipeling much of the time is saved from processing, improves speed of processor a lot. We can divide the pipe into several segments, where each of the segments is dependent on the other. We can add more no of segments as per need but taking care of pipeline hazards.

III. VHDL CODING BASIC

VHDL stands for VHSIC (very high speed integrated circuits) hardware description language. It is used for designing digital circuits. This language describes the architectural, behavioral and physical characteristics of the circuit. VHDL describe the components of the circuit and their interconnections. There are some inbuilt functions in this language which can be used at run time. We can design different modules in this language using Xilinx software. We can also simulate the design using logic sim simulator. Features of Xilinx allow us to make the design more effective. We can also make layout of the design. For a processor design we can design ALU, CU, and Memory etc in different modules then we can add them all and simulate them to find the actual delay. There are three modelling ways in this language-behavioral modelling, dataflow modelling and structural modelling. The execution of any VHDL program allows us to verify the design prior to fabrication. The language is very user friendly. It is very flexible approach for designing digital circuits. There are three steps for designing a model i.e. Design entity, Entity declaration, Architecture body.

RISC Processors:

Some characteristics of RISC are as follows

Reduction of complexity was the main research for RISC.Complex instructions are slower as compared to simpler instructions. Here register to register transfer takes place. Risc with pipeling shows better results.

1. Same length instruction:

Each instruction is the same length so that it may be fetched in a single operation.

2. Machine-cycle instructions:

Most instructions complete in one machine cycle, which allows the processor to regulate several instructions at the same time. This pipelining is a key technique used to speed up RISC machines.

3. Higher speed:

If simpler instructions are there then easy for processor to process them and generating output. Hence speed will increase.

4 Less no of Transistors:

Obviously the complexity has reduced intern no of transistors also reduce, And therefore power will be come.

IV. RISC PROCESSORS

Some characteristics of RISC are as follows
Reduction of complexity was the main research for RISC. Complex instructions are slower as compared to simpler instructions. Here register to register transfer takes place. RISC with pipeling shows better results.

1. Same length instruction:
Each instruction is the same length so that it may be fetched in a single operation.

2. Machine-cycle instructions:
Most instructions complete in one machine cycle, which allows the processor to regulate several instructions at the same time. This pipelining is a key technique used to speed up RISC machines.

3. Higher speed:
If simpler instructions are there then easy for processor to process them and generating output. Hence speed will increase.

4. Less no of Transistors:
Obviously the complexity has reduced intern no of transistors also reduce, And therefore power will be come**.**

Advantages of RISC Processors:

- It improves the speed of the processor.
- It has simpler instruction set.
- It is user friendly.
- By combining pipelining we can obtain better results.
- The instructions executes in single cycle.
- It has large no of general purpose registers.
- Instructions are of same size, regularity is maintained.
- Dedicated load/store instructions move data into and out of the general purpose registers.

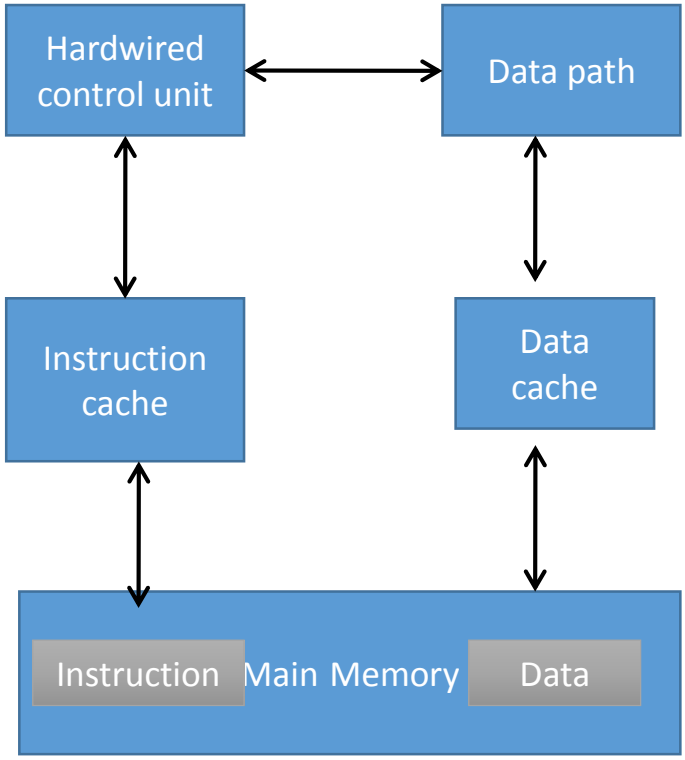


Figure 1: An abstract view of the RISC processor

V. RELATED WORK

Ben Keller, Jaehwa Kwak (2017) This paper presents a RISC-V system-on-chip (SoC) with integrated voltage regulation, adaptive clocking, and power management implemented in a 28 nm fully depleted siliconon-insulator process. A fully integrated simultaneous-switching switched-capacitor DC–DC converter supplies an application core using a clock from a free-running adaptive clock generator, achieving high system conversion efficiency (82%–89%) and energy efficiency (41.8 double-precision GFLOPS/W) while delivering up to 231 mW of power. A second core serves as an integrated power-management unit that can measure system state and actuate changes to core voltage and frequency, allowing the implementation of a wide variety of power management algorithms that can respond at sub microsecond timescales while adding just 2.0% area overhead. A voltage dithering program allows operation across a wide continuous voltage range (0.45 V–1 V), while an adaptive voltage scaling algorithm reduces the energy consumption of a synthetic benchmark by 39.8% with negligible performance penalty.

J.Poornima, G.V.Ganesh (2016) The development of CMOS technology provides very high density and high performance integrated circuits. The performance provided

by the existing devices has created a never-ending greed for increasingly better performing devices. This predicts the use of a whole RISC processor as a basic device by the year 2020. However, as the density of IC increases, the power consumption becomes a major threatening issue along with the complexity of the circuits. Hence, it becomes necessary to implement less complex, low power processor designs. Here in this RISC processor design we mainly concentrate on program counter and ALU.Then this RISC processor is implemented to D.S.P applications like convolution and correlation. In order to employ the processor for signal processing applications, we have integrated a general multiplication in ALU. We can achieve the high speed, low power and area efficient operations by reducing the stronger operations such as multiplication, at the cost of increasing the weaker operations such as addition. 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.

Canturk Isci(2016) Chip-level power and thermal implications will continue to limiters. The gap between average and peak power actually widens with increased levels of core integration. As such, if per-core control of power levels (modes) is possible, a global power manager should be able to dynamically set the modes suitably. This would be done in tune with the workload characteristics, in order to always maintain a chip-level power that is below the specified budget. Furthermore, this should be possible without significant degradation of chip-level throughput performance. We analyze and validate this concept in detail in this paper. We assume a per-core DVFS (dynamic voltage and frequency scaling) knob to be available to such a conceptual global power manager. We evaluate several different policies for global multi-core power management. In this analysis, we consider various different objectives such as prioritization and optimized throughput. Overall, our results show that in the context of a workload comprised of SPEC benchmark threads, our best architected policies can come within 1% of the performance of an ideal oracle, while meeting a given chip-level power budget. Furthermore, we show that these global dynamic management policies perform significantly better than static manage.
This paper presents a RISC-V system-on-chip (SoC) with integrated voltage regulation, adaptive clocking, and power management implemented in a 28 nm fully depleted silicon on- insulator process. A fully integrated simultaneous-switching switched-capacitor DC–DC converter supplies an application core using a clock from a free-running adaptive clock generator, achieving high system conversion efficiency (82%–89%) and energy efficiency (41.8 double-precision GFLOPS/W) while delivering up to 231 mW of power. A second core serves as an integrated power-management unit that can measure system state and actuate changes to core voltage and frequency, allowing the implementation of a wide variety of power management algorithms that can respond at sub microsecond timescales while adding just 2.0% area overhead. A voltage dithering program allows operation across a wide continuous voltage range (0.45 V–1 V), while an adaptive voltage scaling algorithm reduces the energy consumption of a synthetic benchmark by 39.8% with negligible performance penalty.

Ben Keller, Jaehwa Kwak (2017) 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 behavior. For designing any digital system VHDL is the best language. Pipeling is something we come across day to day life. To design a CPU we need instruction set, no of registers and their details, data path 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 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 design and verification of digital circuits at higher register-transfer level of abstraction. It is also used in the verification of analog circuit and mixed-signal circuits, as well as in the design of genetic circuits.

In the design implementation of 16 bit CPU the clock period is highly reduced as 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 behavior. For designing any digital system VHDL is the best language.

Software Used

XILINX'S is truly integrated EDA software encompassing IC designs from concept to completion, enabling chip designers to design beyond their imagination. XILINX'S integrates traditionally separated front-end and back-end chip design into an integrated flow, accelerating the design cycle and reduced design complexities. XILINX'S thus facilitates to vary the parameters of MOSFETS etc so that our study becomes easy.

Methodology

In mathematics, optimization is the selection of a best element under some constraints from some set of available solutions. Optimization problem consists of maximizing or minimizing a real function by systematically choosing input values from within an allowed set and computing the value of the function. The generalization of optimization theory and techniques to other formulations comprises a large area of applied mathematics. More generally, optimization includes finding "best available" values of some objective function given a defined domain or a set of constraints. Types of optimization techniques are discussed below.

VI. CLASSICAL OPTIMIZATION TECHNIQUES

The classical optimization techniques are useful in finding the optimum solution or unconstrained maxima or minima of continuous and differentiable functions. These are analytical methods and make use of differential calculus in locating the optimum solution. The classical methods have limited scope in practical applications as some of them involve objective functions which are not continuous and/or differentiable. Yet, the study of these classical techniques of optimization form a basis for developing most of the numerical techniques that have evolved into advanced techniques more suitable for today's practical problems.

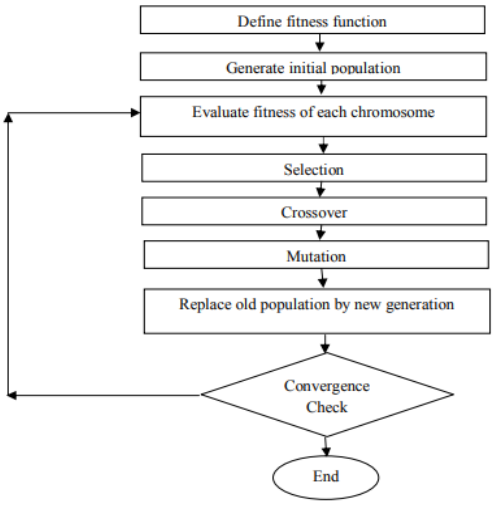
PARAMETERS OF GENETIC ALGORITHMS

A number of parameters control the precise operation of the genetic algorithm. They are:

1. Crossover probability: It is the measure of how often crossover will be performed. If there is no crossover, offspring are exact copies of parents. If there is crossover, offspring are made from parts of both parent's chromosome. If crossover probability is 100%, then all offspring are made by crossover. If it is 0%, whole new generation is made from exact copies of chromosomes from old population. Crossover is made in hope that new chromosomes will contain good parts of old chromosomes and therefore the new chromosomes will be better.
2. Mutation probability: It is the measure of how often parts of chromosome will be mutated. If there is no mutation, offspring are generated immediately after crossover without any change. If mutation is performed, one or more parts of a chromosome are changed. If mutation probability is 100%, whole chromosome is changed, if it is 0%, nothing is changed. Mutation generally prevents the genetic algorithm from falling into local extremes and helps in recovering the lost genetic material. Mutation should not occur very often, because then genetic algorithms would act as to random search.
3. Population size: It is the number of how many chromosomes are present in the population (representing one generation). If there are too few chromosomes, genetic algorithm has few options available for crossover and only a small part of search space is explored. On the counterpart, if there are too many chromosomes in one population then the speed of genetic algorithm slows down.

STEPS IN BASIC GENETIC ALGORITHM

1. [Start] Define the fitness function f(x) according to the problem definition.
2. [Initialise] Generate random population of n chromosomes – each chromosome being the potential solution.
3. [Fitness] Evaluate the fitness f(x) of each chromosome x in the population.
4. [New population]Repeat the following steps to create the new population of chromosomes:
    a. [Selection] Select some parent chromosomes from a population according to their fitness to form mating pool.
    b. [Crossover] Mate the selected chromosomes as per given crossover probability to form new off-springs.
    c. [Mutation]Mutate new chromosomes as per given mutation probability.
    d. [Replace] Replace the old population of chromosomes with the new population.
5. [Convergence check] If the maximum number of generations is reached, then stop, and return the best solution. 6. [Loop] Go to step 3.



Basic flowchart of Genetic Algorithm

VII. CONCLUSIONS

We will be designing the arithmetic unit of the RISC processor using pipelining approach. In the above reviewed papers of the MIPS (Multiprocessor Interlocked processing system) architecture is used for the fixed arithmetic values which will cause the error for the floating numbers. By the use of the floating point unit in the CPU architecture the error will not happen. By using the R-type, I-type, J-type and I/O type of instruction sets the delay can be minimized, power consumption is optimized. As the delay is decreased the speed of the processor is undoubtedly increased. All the modules will be designed in Xilinx using VHDL language.

VIII. REFERENCES

[1]. Sarika U. Kadam, S. D. Mali, "Design of Risc Processor using VHDL", 2016International Journal of Research Granthaalaya, Vol.4 (Iss.6): June, 2016, DOI:10.5281/zenodo.56647.

[2]. Swati Joshi ,Sandhya Shinde,Amruta Nikam, "32-bit pipeline Risc Processor in VHDL using Booth Algorithm ,"International Research Journal of Engineering and Technology(IRJET), e-ISSN: 2395 -0056, Volume: 03 Issue: 04 | April-2016 ,pp.2484-2487.

[3]. Vishwas V.Balpande ,Vijendra P.Meshram,Ishan A. Patilm,Sukeshini N.Tamgadem,Prashant Wanjari, "Design and Implementation of RISC processor on FPGA,"Indian Journal of Advanced Research in computer science and software Engineering, ISSN:2277 128xVol 9(8),Volume 5,Issue 3,March 2015,pp.1161- 1165.

[4]. Soumya Murthy, Usha Verma, "FPGA based Implementation of Power Optimization of 32 Bit RISC Core using DLX Architecture," 2015 International Conference on Computing Communication Control and Automation, DOI 10.1109/ICCUBEA.2015.191

[5]. Mohit N. Topiwala, N. Saraswathi , "Implementation of a 32-bit MIPS Based RISC Processor using Cadence," 2014 IEEE International Conference on Advanced Communication Control and Computing Technologies (ICACCCT), ISBN No. 978-1-4799-3914-5/14/©2014 1EEE.

[6]. Mrs. Rupali S. Balpande,Mrs. Rashmi S. Keote, "Design of FPGA based instruction Fetch and Decode Module of 32-bit (MIPS) processor," International Conference on communication Systems and Network Technologies,DOI:10.1109/CSNT.2011.91,2011.

[7]. Preetam Bhosle, Hari Krishna Moorthy, "FPGA Implementation of low power pipelined 32-bit RISC Processor", International Journal of Innovative Technology and Exploring Engineering (IJITEE), August 2012.

[8]. Sharda P. Katke, G.P. Jain,"Design and Implementation of 5 Stages Pipelined Architecture in 32 Bit RISC Processor", IJETAE, Volume 2. Issue 4 April 2012, pp. 340-346.

[9]. RISC, CISC and Floating point Wikipedia.

[10]. Kui YI, Yue-Hua DING, "32-bit RISC CPU Based on MIPS Instruction Fetch Module Design", 2009 International Joint Conference on Artificial Intelligence, 978-0-7695-3615-6/09, 2009 IEEE.

[11]. Gautham P, Parthasarathy R. Karthi, Balasubramanian. "Low Power Pipelined MIPS Processor Design," in the proceedings of the 2009, l2th international symposium,2009 pp. 462-465.

[12]. Neenu Joseph. Sabarinath. S. "FPGA based Implementation of High Performance Architectural level Low Power 32-bit RISC Core", 2009 IEEE.

[13]. Harpreet Kaur, Nitika Gulati, "Pipelined MIPS With Improved Datapath", IJERA, Vol. 3, Issue 1, January -February 2013, pp.762-765.

[14]. Pejman Lotfi-Kamran. Ali-Asghar Salehpour. AmirMohammad Rahmani. Ali Afzali-Kusha, and Zainalabedin Navabi. "Dynamic Power Reduction of Stalls in Pipelined Architecture Processors", International Journal Of Design, Analysis And Tools For Circuits And Systems. Vol. I, No. I, June 2011.