ISSN No:-2456-2165

# Computing Performance Enhancement of VLIW Architecture Using Instruction Level Parallelism

V. Venkata Nagendra Reddy[1], A. Sudhakar[2], Dr. P. Sivakumar[3]
[1][2]Student, Electronics and CommunicationEngineering, Kalasalingam Academy of Research and Education,Virudhunagar-626126, INDIA
[3]Faculty, Dean student affairs, Kalasalingam Academy of Research and Education, Virudhunagar-626126, INDIA

**Abstract:- Our paper proposes the new method of processor architecture called as VLIW for enhancing the performance of the architecture. VLIW is the complexity architecture because the enormous number of registers, slices, flip flops, counters, operand, ALUs, and MUXs used. The VLIW has the fife stages of pipelines for executing the architecture are (1) fetching the 128-bit instruction memory, (2) decode stage or it is also called as the operands reading stage because the total number of operands are implemented in this stage, (3) execution stage, here the operations with the parallel executions units which has the four operations, (4) memory stage is used for loading or for storing the data from/to the memory and (5) write back stage in this stage the outputs of all the stage is collected and write back into the register file for storing the output values.**

**The whole process of implementation is implemented in the FPGA of the family of Spartan-6 XC6SLX-3CSG324 device. In this proposed architecture the performance of the architecture is increased by reducing the time taken to execute the CPU of Xst completion of the architecture.**

***Keywords:-*** *VLIW Architecture; Vector Processing; Instruction Level Parallelism; FPGA and VHDL Implementation.*

## I. INTRODUCTION

The necessary strategies for gaining the high performance in the architecture is taking the advantage of instruction level parallelism. There is only one simple way to take increment of parallelism is going through with the instruction is through pipelining. If we are not using the pipelining system/technique .the new method of processor architecture called as VLIW for enhancing the performance of the architecture. VLIW is the complexity architecture because the enormous number of registers, slices, flip flops, counters, operand, ALUs, and MUXs used. The VLIW has the fife stages of pipelines for executing the architecture are (1) fetching the 128-bit instruction memory, (2) decode stage or it is also called as the operands reading stage because the total number of operands are implemented in this stage, (3) execution stage, here the operations with the parallel executions units which has the four operations, (4) memory stage is used for loading or for storing the data from/to the memory and (5) write back stage in this stage the outputs of all the stage is collected and write back into the register file for storing the output values

## II. THE ARCHITECTURE OFVECLIW PROCESSOR

VLIW is a load storing architecture with easiest hardware, with minimum length instruction encoding and easy coding generation model. It can support a very minimum addressing models to rectify operands: such as reg, Imm, and displacement addressing modes. In the displacements addressing mode, a constant is additionally extended to form the memory to the scalar register addressed to loading/storing 128-bit data. VLIW architecture has a easy pipeline ISA format, which supports the easy and normal categories of operations such as data transfer, arithmetic, logic, and control unit. According to this there are four types of VLIW instructions are V- elements have the maximum Vector lengths.

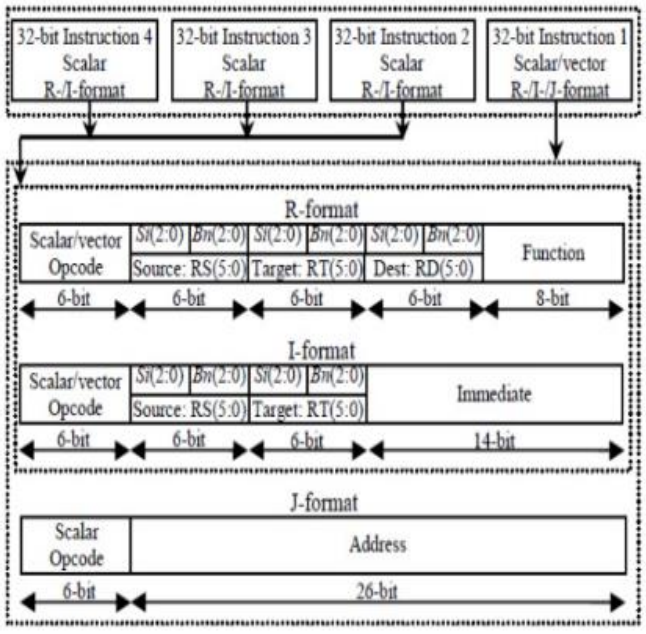

Fig 1:- VLIW ISA formats.

- Scalar- Vector instructions (.sv type),
- Scalar- Scalar instructions (.ss type),
- Vector - Scalar instructions (.vs type),
- Vector Vector instructions (.vv type).

ISSN No:-2456-2165

The ISA format of the VLIW architecture is shown below. The above shown figure is 128bit VLIW architecture ISA format (VLIW[128:1]).There are three formats in the VLIW ISA format are Rformat, I-format J-format. The foremost 32-bit instruction is (VLIW[32:1]) it is a scalar/vector/control instruction. The further instructions are (VLIW[64:33]) (VLIW[96:63]) and (VLIW[128:95]) are must be scalar instructions. Here the control instruction has only one operation. instructions of an application. Here one register file is used for both as multi-scalar and multivector elements. The control units feeds the execution units bit results per clock cycle which comes form the memory system and from the execution units. Even though the percentage of DLP is low, the unified hardware is used for the processing multi-scalar and multi- vector instruction. Comparing the baseline scalar processor there are five stages in the proposed paper are Fetch, decode, execution and write back stages are about four times.

In detailing about architecture pipeline, In the first step the fetching the 128-bit of a VLIW instruction, Second step is about the decoding operands for four individual instructions, the third step is executing the four operation which is decoded in the second step, the fourth step is storing the data in the 128-bit memory, Finally the fifth step is about the writing back the four results to the register file. The VLIW instruction IF/ID pipeline register is loaded with the instruction cache of the fetch from the PC which is pointed. In the decode stage the Control unit is used to fetched VLIW, Which is the Very Long Instruction Word. By using the instruction level parallelism for the processor for the effective method. In parallel by using the different types of operands and it can generate four results in each clock cycle. The four results in the form of 4*32-bit per unit. Final stage is the writeback stage writes into the VLIW register file for storing the output. The 4*32-bits are gotten to fromVLIW register record utilizing 3-bit bank number (Bn) connected with 3-bit start file (Si). 2×4×32bitoperands can be perused and 4×32-bit can bekept in touch with the VLIW register record each clock cycle. RS of every individual guidance in connected to the mux and the output of mux is NPC this is sent to the input of the mux which VLIW register record utilizing 3-bit bank number (Bn) connected with 3-bit start file (Si). 2×4×32bitoperands can be perused and 4×32-bit can bekeptin touch with the VLIW register record each clock cycle. (register destination) of every individual guidance in connected to the mux and the output of mux is NPC this is sent to the input of the mux which is in the Execute stage.

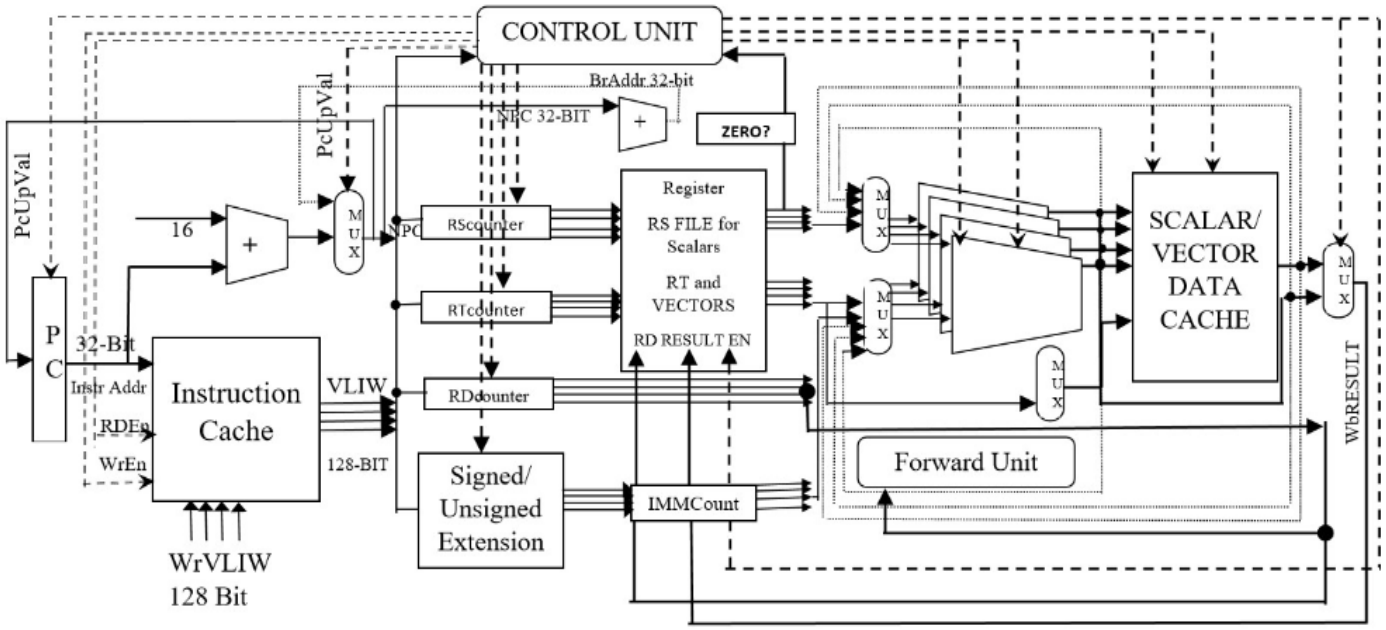


Fig 2:- Block Diagram of VLIW ARCHITECTURE

In this manner, the control unit peruses the RS (register source), RT (register target), and RD the got VLIW just as VLR (vector length register) to create the succession of control signals required for perusing/composing multi-scalar/vector information from/to VLIW register record. The first stage is the Fetch Stage, second one is the Decode Stage, third one of those is Execute Stage, fourth stage of that is Memory Stage, and finally the fifth stage is Writeback Stage. In the first stage there are three modules ,but it happens when it is read enable, and next the Program counter is is located in the decode stage.

The second module is the Decode stage has three components are mainly the control unit, Register file and the hazard detection. The control is the main part which plays an important role for connecting and maintain the certain values as per requirement and the output of program counter is connected to the control unit. The second one is register file is the plays vital role for storing the output and result the entire processor. In the control unit the ALU operations are also done with the 5-bit operation. The third component is the Hazard detection is used to detecting the corrupted interlines of the hardware when it reduces.

ISSN No:-2456-2165

The third module is the VLIW has two components are ALUs and the Forward unit. THE ALUs has a four operands which is declared in the decode stage the output of the RS file of scalar and the RT file of the vector is sent to the input of the MUXs. There are three MUXs used in the execution stage. The output of the two mux is sent to the input of the four ALUs, and the out put of the RD counter is sent to the forward unit. The fourth module is the Memory stage, here the stored component named as data memory. Based on the opcodes used in the processor and the register files used in the architecture destination file is highlighted by RD and RT scalar and vector files. 128-bit data returns from the memory and placed in the writeback stage. If the stored data is from the memory register is return to the Aluout.

The final module is the write back stage, here the scalar and vector results writes into the VLIW register file. It can take results from memory through the multiplexers. All these results sent to the register files which is located in the decode stage.

## III. IMPLEMENTATION OF SOFTWARE/HARDWRAE TOOLS IN VLIW PROCESSOR

The overall design/implementation of the VLIW architecture is implemented by using the VHDL programming language executing in the Xilinx FPGA spartan-6, XC6SLX-3CSG324 device. A single spartan-6 configure the logic blocks comprise two slices, with each containing the four 6-inputs LUTs and flip-flops. Here the spartan-6 family will helps to produce the LUTs as solved either 64x1 or 32x2 RAM distributed. Figure 3 demonstrates the RTL schematic diagram of our proposed VLIW. It is authorized from organizing the VHDL code of the VLIW processor on Xilinx ISE 14.7. Our implementation of the VLIW pipeline consists of five components: The first stage is the Fetch Stage, second one is the Decode Stage, third one of those is Execute Stage, fourth stage of that is Memory Stage, and finally the fifth stage is Writeback Stage. In the first stage there are three modules ,but it happens when it is read enable.

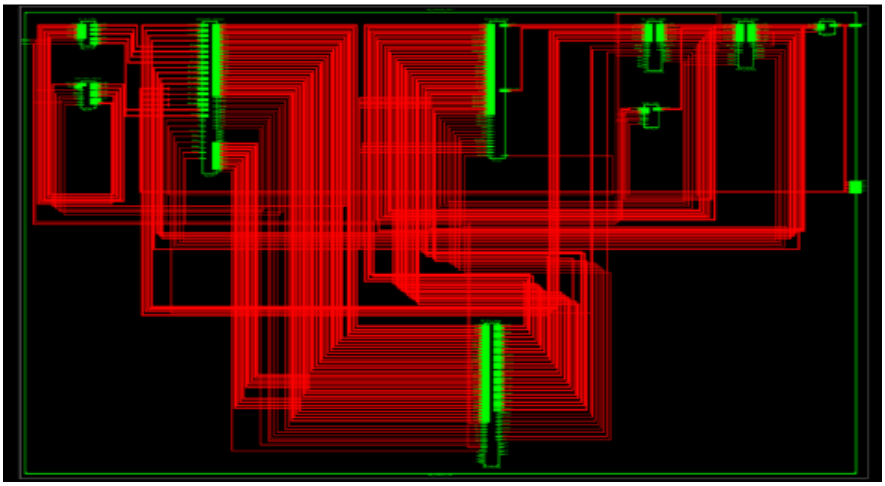

Fig 3:- RTL schematic diagram of the VLIW architecture.

The second module is the Decode stage has three components are mainly the control unit, Register file and the hazard detection. The control is the main part which plays an important role for connecting and maintain the certain values as per requirement and the output of program counter is connected to the control unit. The second one is register file is the plays vital role for storing the output and result the entire processor. In the control unit the ALU operations are also done with the 5-bit operation. The third component is the Hazard detection is used to detecting the corrupted interlines of the hardware when it reduces.

The third module is the VLIW has two components are ALUs and the Forward unit. THE ALUs has a four operands which is declared in the decode stage the output of the RS file of scalar and the RT file of the vector is sent to the input of the MUXs. There are three MUXs used in the execution stage. The output of the two mux is sent to the input of the four ALUs, and the out put of the RD counter is sent to the forward unit. The fourth module is the Memory stage, here the stored component named as data memory. Based on the opcodes used in the processor and the register files used in the architecture destination file is highlighted by RD and RT scalar and vector files. 128-bit data returns from the memory and placed in the writeback stage. If the stored data is from the memory register is return to the Aluout.

The final module is the write back stage, here the scalar and vector results writes into the VLIW register file. It can take results from memory through the multiplexers. All these results sent to the register files which is located in the decode stage.

## IV. RESULT

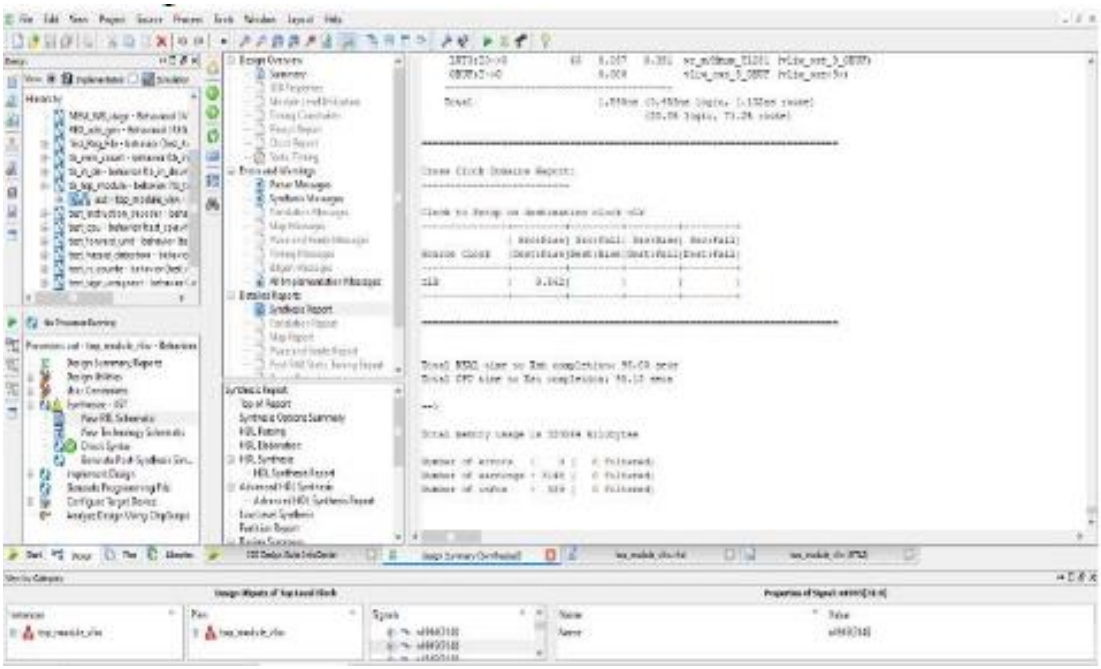

Fig 4

In the above figure the total time to Xst completion is 98.00 sec and the total CPU time to Xst completion is 98.13 sec. This is because the enormous number of register used in the architecture. We reduced the registers used in between the fetch stage and decode stage. Here register files is act as buffers to pass the output came from the previous stage to the next stage. But we can pass the values without using the buffer/register files. We can eliminate the register files by connecting the output values directly to the required input ones.

In the below shown figure the total time taken to Xst completion is 30.00 sec and the total CPU time taken to Xst completion is 30.53 sec. Here these all changes is made to improve the performance of the architecture and reduce the time taken to execute the VHDL code of the architecture and the hardware tool is used to check the code is Spartan-6 which is in the family of FPGA.

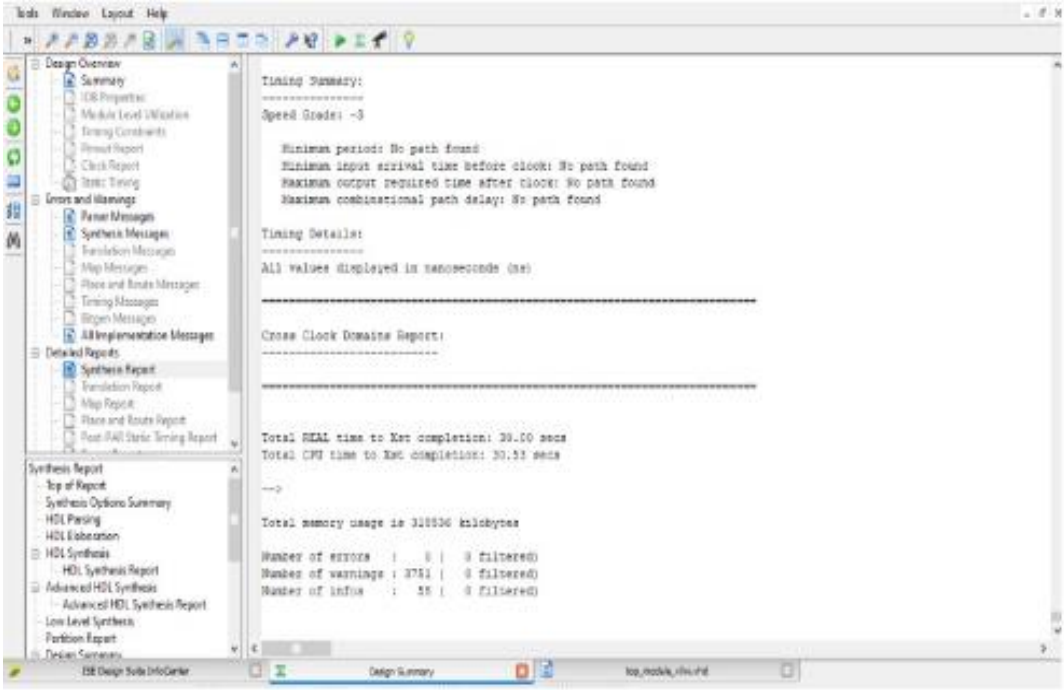


Fig 5

## V. CONCLUSION AND FUTURE WORK

Our paper proposes the new method of processor architecture called as VLIW for enhancing the performance of the architecture. VLIW is the complexity architecture because the enormous number of registers, slices, flip flops, counters, operand, ALUs, and MUXs used. The VLIW has the fife stages of pipelines for executing the architecture are (1) fetching the 128-bit instruction memory, (2) decode stage or it is also called as the operands reading stage because the total number of operands are implemented in this stage, (3) execution stage, here the operations with the parallel executions units which has the four operations, (4) memory stage is used for loading or for storing the data from/to the memory and (5) write back stage in this stage the outputs of all the stage is collected and write back into the register file for storing the output values.

The whole process of implementation is implemented in the FPGA of the family of Spartan-6 XC6SLX-3CSG324 device. In this proposed architecture the performance of the architecture is increased by reducing the time taken to execute the CPU of Xst completion of the architecture. In the future, the performance of our proposed VLIW is used in the scientific calculators and is expandable upto 1064 bites wider and can be used in different applications.

## REFERENCES

[1]. Mostafa I. Soliman A VLIW Architecture for Executing Multi-Scalar/Vector Instructions on Unified Datapath, Computer Science and Information Department, 2013 IEEE.

[2]. J. Mike, Superscalar Microprocessor Design, Prentice Hall (Prentice Hall Series in Innovative Technology), 1991.

[3]. J. Smith and G. Sohi, “The microarchitecture of superscalar processors,” Proceedings of the IEEE, vol. 83, no. 12, pp. 1609-1624, December 1995.

[4]. J. Fisher, “VLIW architectures and the ELI512,” Proc. 10th International Symposium on Computer Architecture, Stockholm, Sweden, pp. 140-150, June 1983.

[5]. J. Fisher, P. Faraboschi, and C. Young, Embedded Computing: A VLIW Approach to Architecture, Compilers and Tools, Morgan Kaufmann, 2004.

[6]. Philips, Inc., An Introduction to Very-Long Instruction Word (VLIW) Computer Architecture, Philips Semiconductors, 1997.