An area efficient and high Throughput 32 bit DSP Processor Design

1Sapana tiwari, 2Prof. Sunil Shah
GGITS, Jabalpur

Abstract: **Proposed thesis work concept comes after thinking and deep study of all these methods of enhancing processor performance and proposing a new DSP microprocessor architecture and it has FSM and some instruction set (big instruction set which better for various applications) and RISC kind of feature like executing every instruction in one cycle. Thesis work has use Xilinx ISE 12.2 ISE software for designing all modules and each module has been tested and verified with all possible instruction which are been supported by proposed design. The proposed architecture of processor requires less area and higher speed as compare with the existing architecture available, it has certain limit that cannot work with medium or sophisticated scale application or it can say it is not suitable for OS based application. As the design core processor is been for small scale non OS based applications which are very common in Microcontroller.**

I-Methodology

As RISC are popular any normally needed where some specific applications are to handles. Initialy for thesis work we have implemented generalized RISC processor with same proposed concept of isolated memories for OPCODE, Immediate Operands and Data as shown in figure 4.1. Later on now this work is specific for DSP application and concerns about execution of fast DSP operations. For that a new Instruction set been developed as compare to our initial work which was using 8085 instruction set. And it is now more efficient because of proposed Reduce (Small and specific) Instruction. This proposed Design is of 16 bit DSP Processor using VHDL, the designed module will be synthesized using Xilinx ISE 9.1i Web pack, and the verification will be done on ISE simulator, and then for validation be the design module will be implemented on Xilinx FPGA (Field programmable Gate Array) Vertex-4 [11].

Figure-4.2 shows proposed architecture, here as can be seen there are two isolated memories MEM1 is further have three different kind first MEM1-1 for signal 1, second MEM1-2 for signal 2 and third MEM1-3 for signal 3. Three signals can be read independently, MEM2 is there for holding the results because the answer can big enough so it can also be divided into two parts MEM2-1 and MEM2-2 for holding LSB and MSB of results respectively.

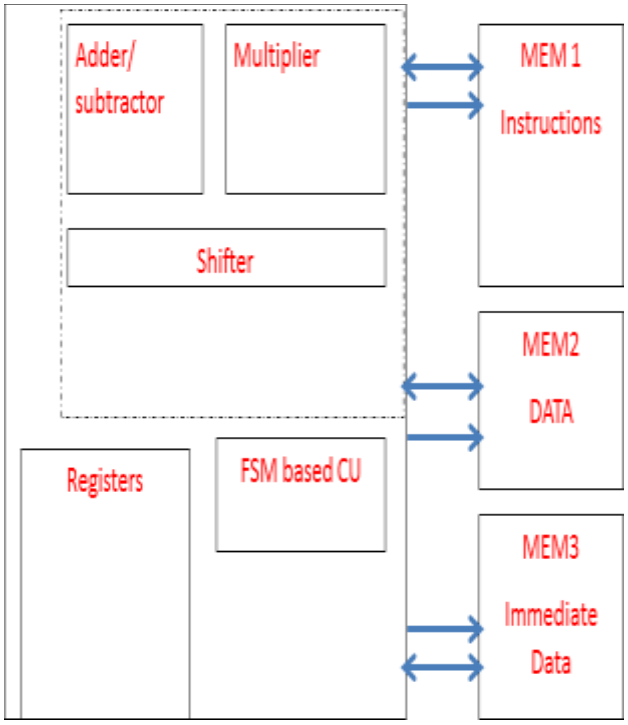


Figure 1: proposed RISC cum DSP processor architecture

Registers are there for holding temporary data during the execution of program. CU is been used for fetching the instruction from MEM2 then extracting OPCODE and operands from instruction, then after decoding the in OPCODE and generate required OPCODE for ALU module, here one can say CU is the master for ALU and it give order to the ALU. CU also read the signals from MEM1 as per the decoded OPCODE and CU also read the results generated from the ALU and also write back it into the MEM2.

Instruction format and Opcode's: ALU has three major modules Adder-Subtractor, shifter and Multiplier, the multiplier user for the multiplication of two numbers, DSP generally deals with floating numbers so adder, subtractor and multiplier should be floating in nature The floating numbers in proposed work has 6 binary places, let's have an example if we want to write 13.75 it would be 0000001101.110000.

Total size of is of 32 bit

OPCODE (8 bit)	Pointer of signal one (4 bit)	Pointer of signal two (4 bit)	Pointer of signal three (4 bit)	Shifting count of signal one (4 bit)	Shifting count of signal two (4 bit)	Shifting count of signal three (4 bit)

OP(7)	OP(6)	OP(5)	OP(4)	OP(3)	OP(2)
0 if single arithmetic operation else 1	1 if Multiplication required else 0	1 if Addition required else 0	1 if Subtraction required else 0	1 if signal shifting required else 0	0 right shiftin 1 if left shifting

Figure 2 The New Instruction format

II-Results

Figure shown below shows the RTL (Register transfer Logic) schematic/ technological Schematic top of proposed module here:-

clk is the clock input signal
rst is the reset input signal

enb is the enable input signal
dout1 and dout2 are of 16 bit each output data, dout2 MSB 16 bit when multiplication done and dout1 is the 16 bit LSB output
addrw if 4 bit address of output memory
rw1 and rw2 are the output of control for write operation

The signal1, signal2 and signal3 are already inside the module in the form of memory and 32 bit instruction is also inside. The extended RTL and Technological view are shown in Appendix-A.

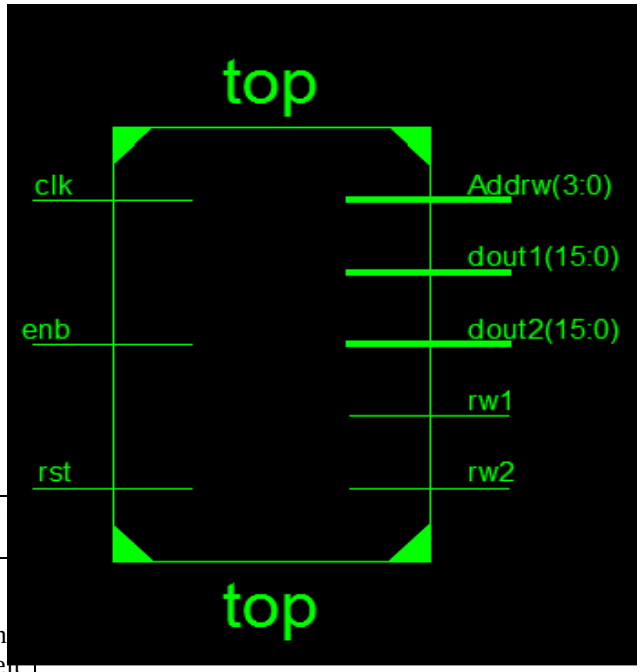


Figure 3: RTL/Technological schematic TOP of proposed processor

Simulation result: simulation is been observe for each scenario here table shown below shows the instruction which is been tested for the three signals below

S1(n)=[0 0 0 0 0 6 1 9 2 8 0 0 0 0 0 0]
S2(n)=[0 0 0 0 0 6 1 9 2 8 0 0 0 0 0 0]
S3(n)=[0 0 0 0 0 6 1 9 2 8 0 0 0 0 0 0]

	Instruction	HEX	Meaning
1	MUL S1(n),S2(n)	40678000	Multiplication of two signal S1(n) and S2(n)
2	MAD S1(n),S2(n),S3(n)	E0678000	Multiplication of two signal S1(n) and S2(n) than addition with S3(n)
3	MAS S1(n),S2(n),S3(n)	D0678232	Multiplication of two signal S1(n) and S2(n) than subtraction of S3(n)
4	MRS S1(n-2),S2(n)	48678200	Multiplication of two signal S1(n-2) and S2(n)
5	MLS S1(n),S2(n+3)	4C678030	Multiplication of two signal S1(n) and S2(n+3)

6	ADD S1(n),S2(n)	20678000	Addition of two signal S1(n) and S2(n)
7	ARS S1(n-2),S2(n)	28678200	Addition of two signal S1(n-2) and S2(n)
8	ALS S1(n+1),S2(n)	2C678102	Addition of two signal S1(n+1) and S2(n)
9	SUB S1(n),S2(n)	10678000	Subtraction of two signal S1(n) and S2(n)
10	SRS S1(n-2),S2(n-3)	18678232	Subtraction of two signal S1(n-2) and S2(n-3)
11	SLS S1(n),S2(n)	1C678002	Subtraction of two signal S1(n) and S2(n)
12	RS S1(n-2)	08678200	Right shift of signal S1(n-2)
13	LS S1(n+2),S2(n+3),S3(n+2)	0C678232	Left shift of signal S1(n+2),S2(n+3),S3(n+2)
14	MAR S1(n),S2(n-3),S3(n-1)	E8678031	Multiplication of S1(n) and S2(n-3) than Addition of S3(n-1)
15	MAS S1(n),S2(n+3), S3(n+2)	EC678032	Multiplication of S1(n) and S2(n+3) than addition of S3(n+2)
16	MSR S1(n),S2(n-3),S3(n)	D8678030	Multiplication of S1(n) and S2(n-3) than subtraction of S3(n)

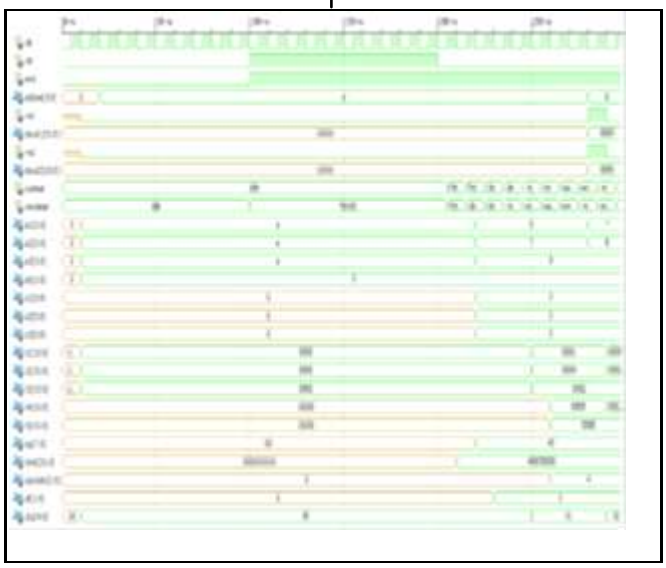
Table 1 the Instructions for simulation

To understand the wave for we requires to know the importance of each signal shown in it

1. Clk: is the clock signal and for testing it is been taken of 50 Mhz (20 ns clock period)
2. Rst: is the reset signal for resetting the signal addresses, initial FSM state and other control signals.
3. Enb: enable signal for enabling the FSM of design
4. Addrw: is the address where output signal to be write after execution
5. Rw1/Rw2: are the write into output memory as outputs can be of possible (when multiplication involved) 32 bits so two 16 bit word size memory required.
6. Dout1/dout2: are the data for two output memories
7. Curstat and nextstat: shown the current and next state of melay type FSM which is been used in our design as control unit
8. a1/a2/a3: are the address of input signal zero pointer in memories
9. a5 is the address of instruction in instruction memory
10. s1/s2/s3: are the shifting in the input signals requires in instruction.
11. t1/t2/t3: are the input signal at the shifted address
12. t4/t5: are the data came from ALU after execution
13. op: is the opcode from the instruction
14. Tinst: it holds the instruction read from instruction memory
15. Opcode: is the opcode of ALU
16. Sf: it is two bit value from op(3:2) it decide left or right shifting if sf="10" right shifting and if sf="11" than it there will be left shifting
17. Ck1: it holds the address of Instruction memory

The total time requires for single instruction to complete is 600 ns if single clock pulse of 20 ns time taken
Now the instruction from table can be understand properly

Instruction 1: MUL S1(n),S2(n)
S1(n)=[6 1 9 2 8]
S2(n)=[6 1 9 2 8]
Expected Result: [0 6 9 18 16 0]



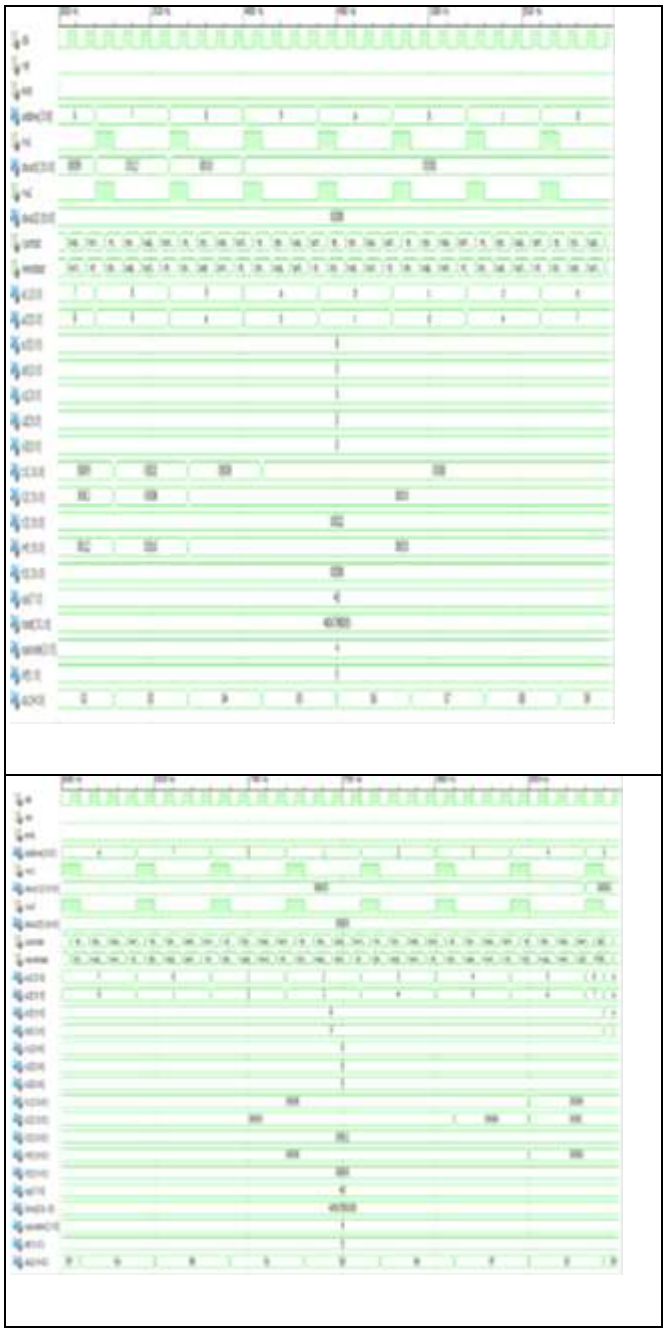


Figure 4 Simulation for test Instruction-I

Expected Answer found in simulation and been highlighted with red dashed line

Synthesis Results: figure shown below shows a synthesis result which is been generated using Xilinx ISE 12.2i.

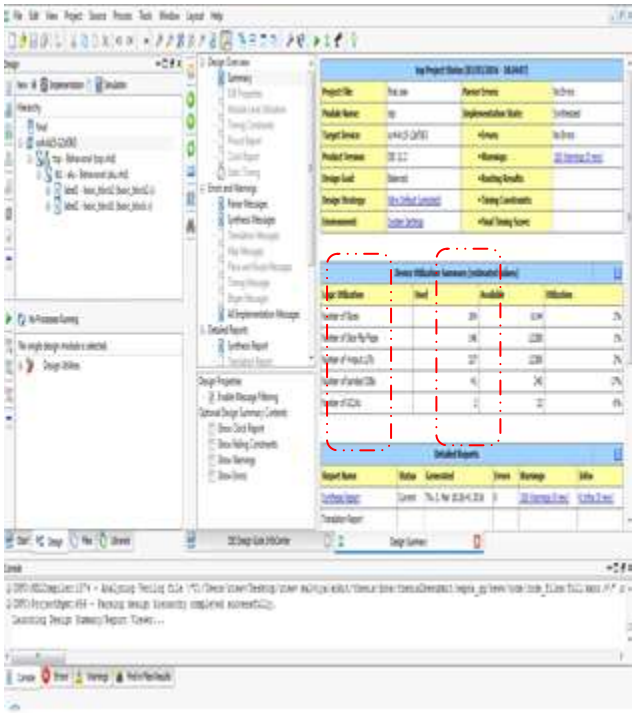


Figure 5: synthesis result of proposed design

Target FPGA family Vertex	
Number of slice	184
Number of LUT	329
Number of slice flip flop	143
Logical time required	3.683 ns
Max freq.	271.517 Mhz

Table 2 The Area and timing results observed for the proposed design

For the target FPGA platform Vertex-4 the result is been observe and synthesis the Slice of Vertex-4 have 2 LUT and one Flip flop and LUT are the programmable circuits which can develop digital circuits of any 4 input one output Sum Of Product equation. Hence one can consider number of slice and LUT as area required for our design. The logical time required for the proposed design is 3.683 ns it is the performance parameter for the proposed work.

III- Conclusion

The proposed work is a new RISC architecture based DSP processor and it can be concluded on behalf of observed results. proposed architecture is best among the existing

work in terms of Area and speed both. The work is a FSM based design and implemented and verified on Xilinx EDA tool. The verification is done with the help ISE simulator of Xilinx and tested for every OPCODE table 2.

This thesis gives a clear statement that FPGAs are a good candidate for efficient implementation of proposed multiplication applications. We hope that there will be some more proposals with higher level of abstraction (e.g vedic Processor, Vedic Controller, Vedic DSP processor etc.) and implementations of some other algorithms in the near future.

References

[1] Khoi-Nguyen Le-Huu, RTL implementation for a specific ALU of the 32-bit VLIW DSP processor core, *Univ. of Inf. Technol., Ho Chi Minh City, Vietnam*, International Conference on Advanced Technologies for Communications (ATC), 2014, DOI: 10.1109/ATC.2014.7043417,IEEE, 2014

[2] Khoi-Nguyen Le-Huu, Towards a RISC instruction set architecture for the 32-bit VLIW DSP processor core, Region 10 Symposium, DOI: 10.1109/TENCONSpring.2014.6863068, 2014, IEEE

[3] Aneesh.R, Jiju.K,Design of FPGA based 8-bit RISC controller IP core using VHDL, India Conference (INDICON), 2012 Annual, DOI: 978-1-4673-2272-0/12 IEEE, 7-9 Dec. 2012,

[4] Ryszyard gal, adma golda, maciej frankiewics, andrzej koz, FPGA implementation of 8 bit RISC Microcontroller for Embedded systems, MIXDES 2011, technical university of Lods,

[5] Ranganadh Narayanam, Artyom M. Grigoryan, Parimal A. Patel, Bindu Tushara D., Implementation and performance evaluation of paired transform based Faster FFT: Grigoryan FFT on Xilinx FPGAs and TMS DSPs using MATLAB: SIMULINK and CC Studio, International Journal of Scientific & Engineering Research, Volume 4, Issue 8, August-2013

[6] Sunitha M S, Bharat G Hegde, Deepaka kumar N Hegde, design and comparison of risc processors using different alu architectures, international Journal of Innovative Research in Science, Engineering and Technology Vol. 2, Issue 7, July 2013

[7] Mrs. Rupali S. Balpande., Mrs.Rashmi S. Keote., Design of FPGA based Instruction Fetch & Decode Module of 32-bit RISC (MIPS) Processor. 2011 International Conference on Communication Systems and Network Technologies, IEEE

[8] Yanfen Chen,Wuchen Wu,Ligang Hou,Jie Hu, VLSI & Integrated System Lab. Design and Implementation of 8-bit RISC MCU, Microelectronics and Electronics (PrimeAsia), 2010 Asia Pacific Conference on Postgraduate Research, 22-24 Sept. 2010, IEEE

[9] Tomas Balderas-Contreras, Rene Cumplido, Claudia Feregrino-Uribe, On the design and implementation of a RISC processor extension for the KASUMI encryption algorithm, Computers and Electrical Engineering 34 (2008) 531–546, ScienceDirect and Elsevier

[10] Chien-Hsuan Wu, Chin-Yu Huang , and Jun-Ru Chang, Application-Specific RISC Architecture for ITU-T G.729 Decoding Processing, TENCON 2006. 2006 IEEE Region 10 Conference, 14-17 Nov. 2006

[11] http://www.xilinx.com/support.html

[12] D. Seal, ARM Architecture Reference Manual. 2nd Edition, Addison- Wesley 2001.

[13]M. K. Jain, M. Balakrishnan, and A. Kumar, "ASIP Design Methodologies: Survey and Issues," Proceeding of Fourteenth International Conference on VLSIDesign, pp. 76-8 1, Jan. 2001.

[14] K. Wakabayashi, and T. Okamoto, "C-Based SoC Design Flow and EDA Tools: An ASIC and System Vendor Perspective," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Volume 19, No. 12, pp. 1507-1522, Dec. 2000.

[15] M. Gschwind, "Instruction set selection for ASIP design," Proceeding of the Seventh of International Workshop on Hardware/Software Codesign, pp. 7-1 1, May 1999.

[16] I. H. Jeng, F. Lai, and Y. D. Tseng, "FACE: Fine-tuned Architecture Codesign Environment for ASIP Development," Design Automation for Embedded Systems, Kluwer Academic Publishers, Vol. 4, No. 4, pp. 329-25 1, Oct. 1999.

