A design for DSP processor for Transform applications

Abhishek Thakur, *M. tech. scholar, E&C dept., SRIT, Jabalpur*
Vivek Dubey, Asst.Prof, E&C dept., SRIT, Jabalpur
Prof. Ravimohan, HOD, E&C dept., SRIT, Jabalpur

Abstract: **As known any microprocessor is a general purpose IC (Integrated Circuit) which simply follow the instructions given to it, and the instructions set for the different microprocessors are different and designed such a way that it and perform required type for computations. Proposed paper work is a new RISC processor architecture for DSP and it has FSM and new instruction set and RISC feature like executing every instruction in one cycle. Paper 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 for processor requires less area and higher speed as compare with the other existing RISC architecture available.**

Keywords:-**IC, CACHE memory, ISE, CISC, FPGA, microcode, RAM, op-code, RISC, SHARC.**

I. INTRODUCTION

A Digital Signal Processor (DSP) is a microprocessor particularly to process the digital signals. It has a specialized architecture which is perfect for the fast operational requirement for digital signal processing. A Digital Signal Processor (DSP) is particularly for those applications that can't tolerate delays because the main feature for DSP is to process the data in real time. Digital Signal Processors take a digital signal and improves the quality for that signal. For ex. it make the sound very clear for that digital signal, gives the faster data or sharper images. Digital Signal Processors use that type for signals that have been digitized like video, voice, audio, temperature or position signals etc and then manipulate mathematically. To perform these mathematical functions rapidly DSP is designed. So the information contained in signals can be displayed or converted to another type for signal after the process for the signals.
There are many different kinds for programmable digital signal processors like image signal processor, radar processor, earthquake measure processor, pixel processor, piccolo processor ARM versatile cortex processor.

As RISC are popular any normally needed where some specific applications are to handles. Earlier for thesis work we have implemented generalized RISC processor with same proposed concept for isolated memories for OPCODE, Immediate Operands and Data as shown in figure 1. Later on now this work is specific for DSP application and concerns about execution for fast DSP operations. For that a new Instruction set been developed as compare to our earlier work which was using 8085 instruction set. And it is now more efficient because for proposed Reduce (Small and specific) Instruction set.

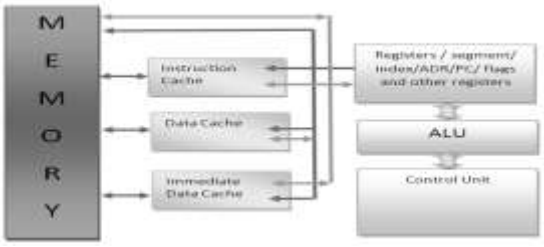


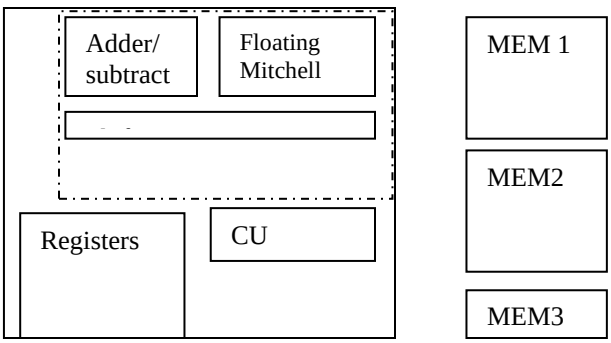
Figure 1: Earlier work

II. PROPOSED ARCHITECTURE

This proposed Design is for 16 bit DSP Processor using Verilog HDL, 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-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 for results respectively.

Figure 2: proposed RISC cum DSP processor architecture



Registers are there for holding temporary data during the execution for 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.

CU is FSM based design for maintaining the synchronization between all modules explains above. Paper work also proposed a new RISC Instruction set for DSP application along with its OPCODE is been show in table 1.

ALU has three major modules Adder-Subtractor, shifter and Multiplier, the multiplier user for the multiplication for two numbers if Iteration based Mitchell method as know DSP generally deals with floating numbers so adder, subtractor and multiplier should be floating in nature and Mitchell multiplication is good for dealing with floating numbers. The floating numbers in proposed work has 6 binary place, let's have an example if we want to write 13.75 it would be 0000001101.110000.

Multiple operation + multiplication + add + sub + (0 no shift+1 shift)+(0 right shift one,1left shift one)			
OPCODE	**HEX**	**Operation**	**Mnemonics**
0100_00XX	40	Multiplication only	MUL
1110_00XX	E0	Multiplication and addition	MAD
1101_00XX	D0	Multiplication and subtraction	MAS
0100_10XX	48	Multiplication and shift right signal one	MRS
0100_11XX	4C	Multiplication and shift left signal one	MLS
0010_00XX	20	Addition only	ADD
0010_10XX	28	Add with right shift signal one	ARS
0010_11XX	2C	Add with left shift signal one	ALS
0001_00XX	10	Subtraction only	SUB
0001_10XX	18	Sub with right shift signal one	SRS
0001_11XX	1C	Sub with left shift signal one	SLS
0000_10XX	08	Shifting only right signal one	RS
0000_11XX	0C	Shifting only left signal one	LS
1110_10XX	E8	Multiplication addition right shift signal one	MAR
1110_11XX	EC	Multiplication addition left shift signal one	MAS
1101_10XX	D8	Multiplication subtraction right shift signal one	MSR
1101_11XX	DC	Multiplication subtraction left shift signal one	MSL

Table 1: OPCODE for proposed design

III. RESULT & SIMULATION

Earlier work was a generalized RISC processor and table 2 shown below shoes its results.

Logic Utilization	**Used**
No. for slice registers	324
No. for slice LUT's	1271
No. for fully used bit slices	214
No. for bonded IOB's	88
No. for block RAM/FIFO	1

Table 2: results observed in Previous generalized RISC processor

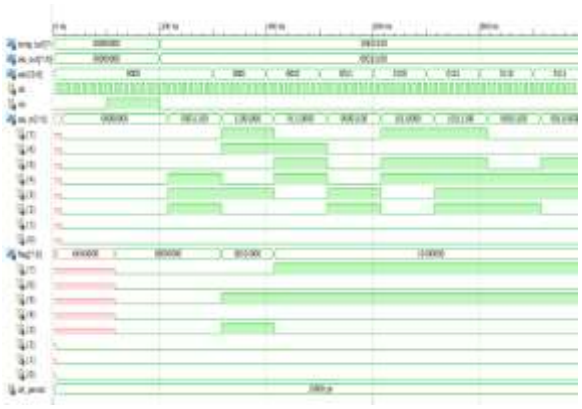
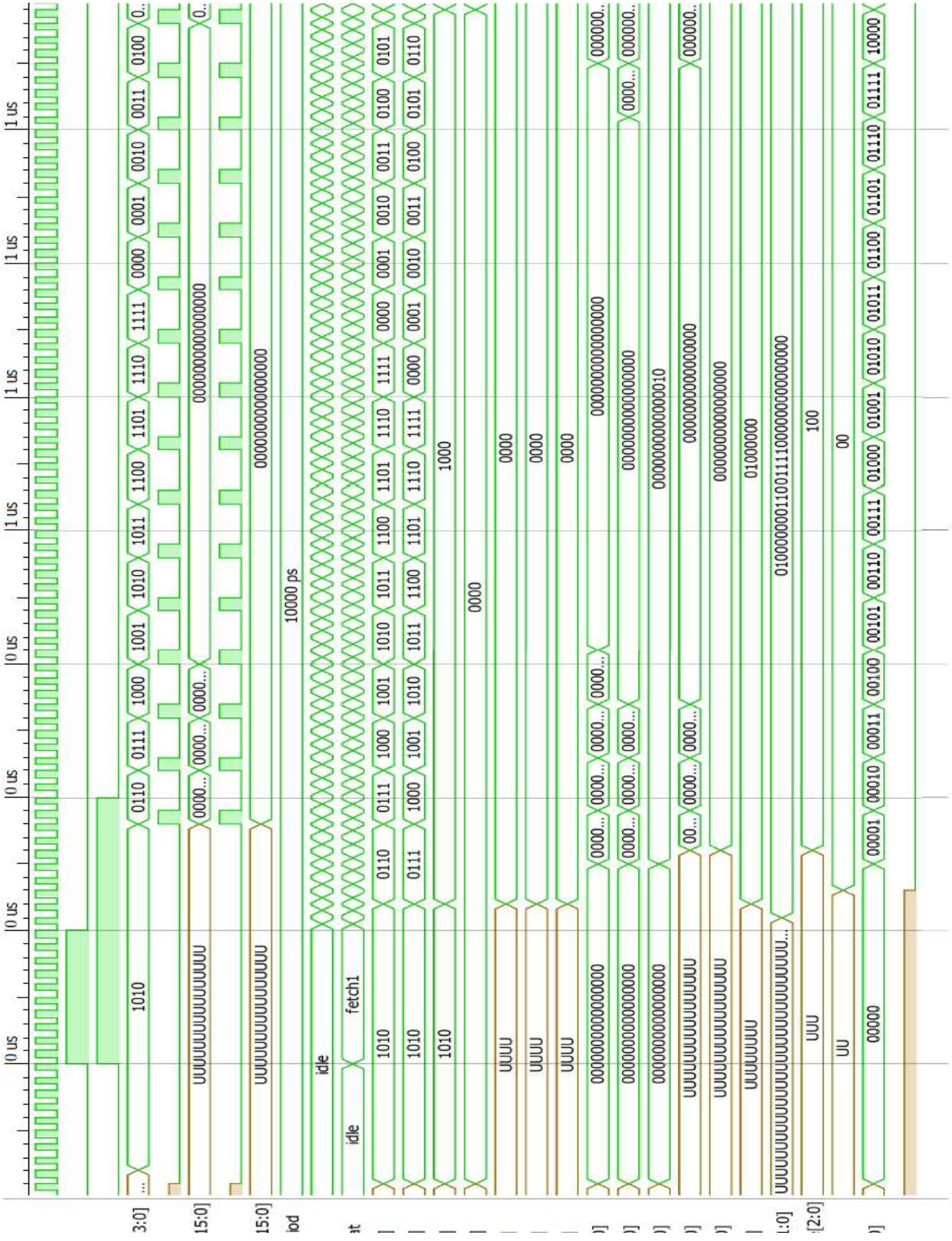


Figure 3: simulation for previous work



The Proposed core architecture has been designed and simulated with the help for Xilinx ISE software. The result is as shown below.

Target FPGA family Vertex 4	
Number for slice	184
Number for LUT	329
Number for slice flip flop	143
Logical time required	3.683 ns
Max freq.	271.517 Mhz

Table 2: Logic Utilization

Table 2 shown above is the observed results for the Proposed DSP processor.

Target FPGA family Vertex family	Proposed	Base 2	Base 4	Base 6	Base 8	Base 9
Number for slice	184	-	-	-	428	448
Number for LUT	329	-	-	1462	-	-
Number for slice flip flop	143	-	-	-	-	-
Logical time required	3.683 ns	-	-	-	-	-
Max freq.	271.517 Mhz	13 Mhz	253.8 Mhz	-	-	96.33 Mhz

Table 3: Comparative Analysis

Table 3 above is the comparative results for proposed work all other existing work, we have took total 10 research papers from various international journals as base work shown in references before starting the actual work. Table above shows comparative results with five research papers.

Figure 4 above shows the simulation observed for the proposed work and it shows execution for MUL instruction for instruction set with two signals having 5 bit length and having different starting positions.

IV. CONCLUSION

The proposed work is a new RISC architecture based DSP processor and it can be concluded on behalf for observed results in table 2 and 3 that proposed architecture is best among the existing work in terms for 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 for Xilinx and tested for every OPCODE for table 1.

REFERENCES

[1] Amit Kumar Singh Tomar, Rita Jain, 20-Bit RISC & DSP System Design in an FPGA, publication in Computing in Science and Engineering,DOI journal, 2013 IEEE

[2] Tasnim Ferdous Design, Synthesis and FPGA-based Implementation for a 32-bit Digital Signal Processor, International Journal for Scientific & Engineering Research, Volume 3, Issue 7, July-2012

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

[4] Aneesh.R, Jiju.K,Design for FPGA based 8-bit RISC controller IP core using VHDL, India Conference (INDICON), 2012 Annual, IEEE, 7-9 Dec. 2012,

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

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

[7] Mrs. Rupali S. Balpande., Mrs.Rashmi S. Keote., Design for FPGA based Instruction Fetch & Decode Module for 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 for 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 for 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

