Volume 5, Issue 1, January 2015 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Design and Implementation of 16 Bit Processor on FPGA

Vishwas V. Balpande*, Abhishek B. Pande, Meeta J. Walke,
Bhavna D. Choudhari, Kiran R. Bagade
Electronics Engineering, DBACER,
Nagpur, Maharashtra, India

Abstract— Thispaper presents the designing and implementation of 16 bit Processor using VHDL (very high speed integrated circuit hardware description language) on FPGA (Spartan-6 kit). The CPU, shifter, comparator, control unit and memory are integrated in proposed processor. The processor has 16 bit arithmetic and logical instruction set analogous to 8085 and also includes multiplication and division instruction. The control unit generates all the control signals needed to control the coordination among the entire component of the processor. All the modules in the design are coded in VHDL (very high speed integrated circuit hardware description language) to ease the description, verification, simulation and hardware implementation. The design entry, synthesis, and simulation of processor are done by using Xilinx ISE 13.2 software and implemented on XC6SLX16-2CGS324 Spartan-6 FPGA device.

Keywords— Register transfer level (RTL), Veryhigh speed integrated circuit hardware description language (VHDL),Arithmetic logic unit (ALU), Central processing unit (CPU), Field programmable gate array (FPGA).

I. INTRODUCTION

Processors are divided into 3 categories 8-bit, 16-bit and 32-bit processor, depending upon the demand ofperformance, cost, power and programmability. 8-bit processors have extreme low cost and consume less powerfor simple control system. In contrast to 8-bit, 32-bit processors have high programmability, high performanceand are widely used in cellular phone and PDA that need high computation but it has high power consumption.On the other hand 16-bit processors have high performance and power than 8-bit processor and low powerconsumption than 32-bit processor. They are often used in 16-bit applications such as disk driver controller,cellular communication and airbags, etc.

The 16-bit fully functional single cycle processor is applicable for real tasks and also used for assembly languageprogramming. The processor is developed using top-down design approach. All the modules of processor as in Fig. 1 are designed using VHDL and then structured to get the top level design of processor.

As the programmable logic technology hasdeveloped rapidly, it is feasible that microprocessors are implemented using Field Programmable Gate Array (FPGA). The top level design of processor has been implemented on Spartan 6 FPGA evaluation development board using Xilinx 13.2.

II. PROCESSOR DESIGN

The processor contains a number of basic modules as in Fig. 1. There is a registerarray of eight 16-bit registers, an ALU (Arithmetic Logic Unit), a shifter,a program counter, an instruction register, a comparator, an address register,and a control unit which controls all the processor modules through different control signal. All of these units communicate through a common,16-bit tristate data bus.

III. PROCESSOR TOP-LEVEL DESIGN

The top-level design consists of the processor block and a memory blockcommunicating through a bidirectional databus, an address bus, and a fewcontrol lines as in Fig. 2. The processor fetches instructions from the external memoryand executes these instructions to run a program. These instructions arestored in the instruction register and decoded by the control unit. The control unit causes the appropriate signal interactions to make the processor unit execute the instruction.

If the instruction is an addition of two registers (ADD R1, R2), the control unit would cause the first register value to be written to register OpReg for temporary storage. The second register value would then be placed on the data bus. The ALU would be placed in add mode and the result would be stored in register OutReg. Register OutReg would store the resulting value until it is copied to the final destination.

When executing an instruction, a number of steps take place. The program counter holds the address in memory of the current instruction. After an instruction has finished execution, the program counter is advanced to where the next instruction is located. If the processor is executing a linear stream of instructions, this is the next instruction. If a branch was taken, the program counter is loaded with the next instruction location directly.

The control unit copies the program counter value to the address register, which outputs the new address on the address bus. At the same time, the control unit sets the R/W (read write signal) to a ‘0’ value for a read operation and sets signal VMA (Valid Memory Address) to a ‘1’, signaling the memory that the address is now valid. The memory decodes

the address and places the memory data on the data bus. When the data has been placed on the data bus, the memory has set the READY signal to a '1' value indicating that the memory data is ready for consumption.
The control unit causes the memory data to be written into the instruction register. The control unit now has access to the instruction and decodes the instruction.

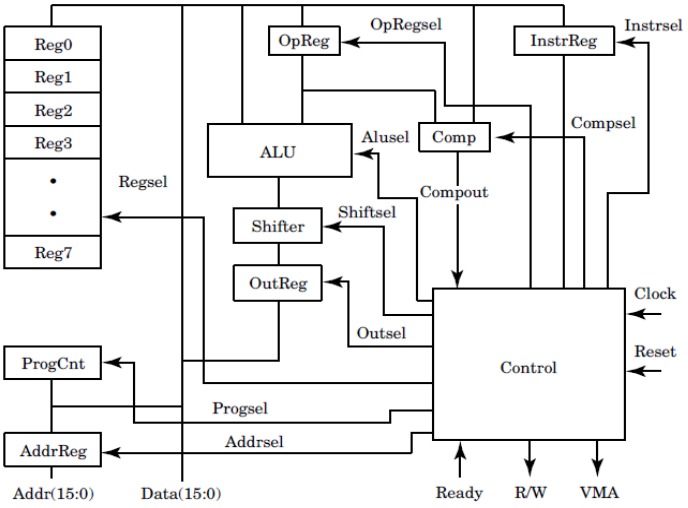

Fig.1 Processor Block Diagram

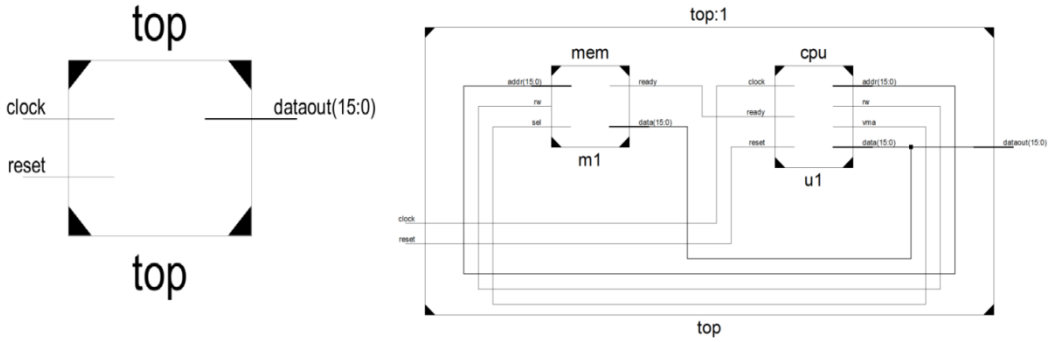

Fig.2 RTL View of Processor's Top Level

IV. INSTRUCTION SET

Instructions can be divided into a number of different types as follows:

- Load—These instructions load register values from other registers, memory locations, or with immediate values given in the instruction.
- Store—These instructions store register values to memory locations.
- Branch—These instructions cause the processor to go to another location in the instruction stream. Some branch instructions test values before branching; others branch without testing.
- ALU—These instructions perform arithmetic and logical operations such as ADD, SUBTRACT, INC, DEC, MUL, DIVISION, OR, AND, XOR and NOT.
- Shift—These instructions use the shift unit to perform shift operations on the data passed to it.

Instructions share common attributes, but come in a number of flavors.Example of instructions is shown in Fig. 3. All instructions contain the opcode in the five most significant bits of the instruction. Single-word instructions also contain two 3-bit register fields in the lowest 6 bits of the instruction. Some instructions, such as INC (Increment), only use one of the fields, but other instructions, such as MOV (Move), use both register fields to specify the From register and the To register. In double-word instructions, the first word contains the opcode and destination register address, and the second word contains the immediate instruction location or data value to be loaded.

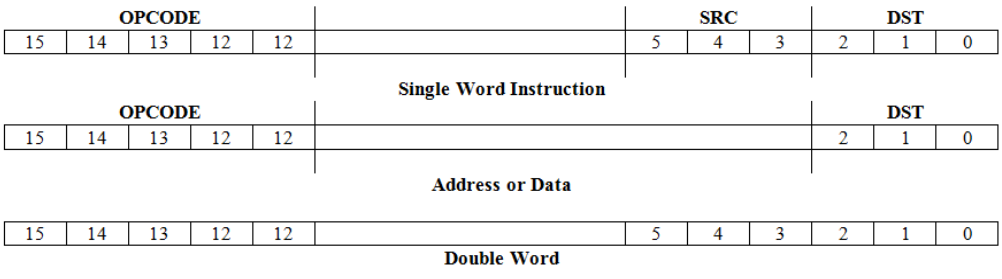

Fig.3 Instruction Format

V. ARITHMETIC & LOGIC UNIT

Arithmetic and Logic Unit is very important module of CPU which performs arithmetic& logical operationson one or more input busses.In general, the ALU includes storage places for input operands, operands that are being processed, the accumulated result (stored in accumulator), and shifted results. Fig. 4 shows the RTL view of ALU and it simulation results.Inputs 'a' and 'b' are the two input busses upon which the ALU operationsare performed. Output bus 'c' returns the result of the ALU operation. Input 'sel' determines which operation is performed as specified in Table I.

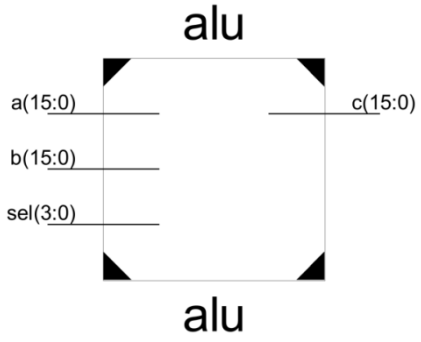

Fig. 4 RTL view of ALU

TABLE I

Sel Input	Operation
0000	C = A
0001	C = A AND B
0010	C = A OR B
0011	C = NOT A
0100	C = A XOR B
0101	C = A + B
0110	C = A - B
0111	C = A + 1
1000	C = A – 1
1001	C = 0
1010	C = A * B
1011	C = A / B

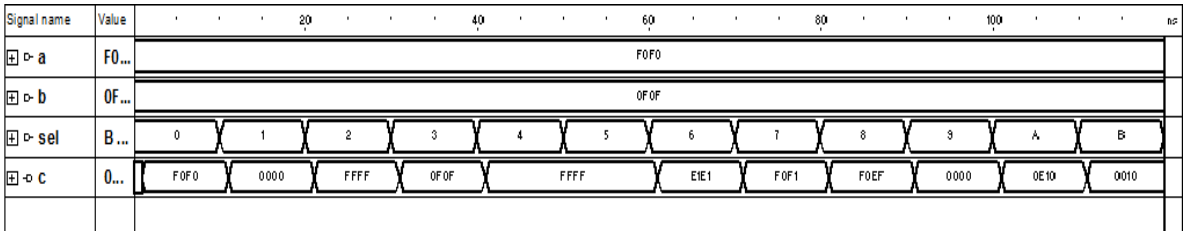

Fig. 5 Simulation Results of ALU

VI. COMPARATOR

Fig. 6 shows RTL view of comparator which compares two values and returns either a '1'or '0'depending on the type of comparison requested given on 'sel' input given in Table II and the values being compared.

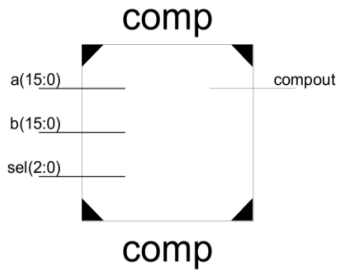

Fig. 6 RTL view of Comparator

TABLE II

Sel Input	Operation
EQ	Compout = 1 when a equals b
NEQ	Compout = 1 when a is not equal to b
GT	Compout = 1 when a is greater than b
GTE	Compout = 1 when a is greater than or equal to b
LT	Compout = 1 when a is less than b
LTE	Compout = 1 when a is less than or equal to b

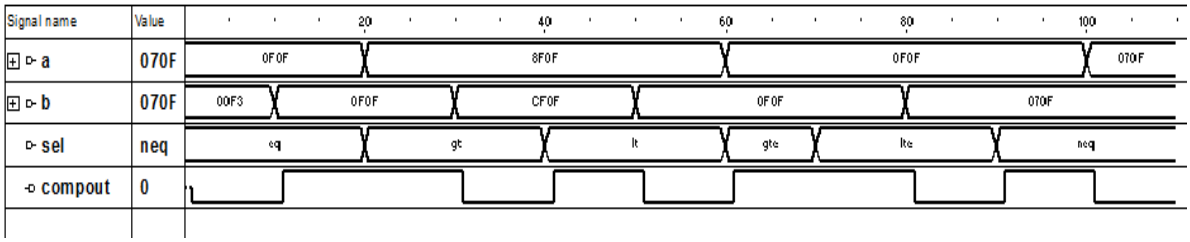

Fig. 7 Simulation Results of Comparator

VII. SHIFTER

The shift entity is used to perform shifting and rotation operations within the CPU. The shift entity has a 16-bit input bus, a 16-bit output bus, and a 'sel' input that determines which shift operation to perform. As can be seen by the figures, the shift entity can perform a shift left, shift right, rotate left, and rotate right operation. One operation that is not shown by the figures is a pass through operation in which all input bits are passed through to the output unchanged.

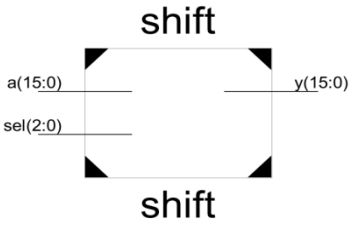

Fig. 8 RTL view of Shifter

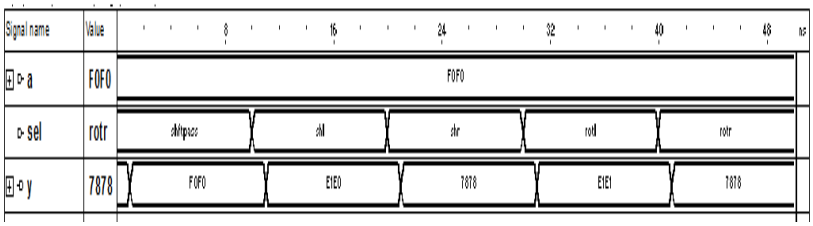

Fig. 9 Simulation Results of Shifter

VIII. CONTROL UNIT

The Control Unitentity provides the necessary signal interactions to make thedata flow properly through the CPU and perform the expected functions. Control Unit consist of a state machine that causes all appropriate signalvalues to update based on the current state and input signals and producea next state for the state machine. The control symbol has only a few inputs, but a lot of outputs. Thecontrol block provides all of the control signals to regulate data traffic forthe CPU.

Control unit composed of controller, program counter,instruction register, and multiplexer. The controller provides thenecessary signal interaction to make the data flow through theprocessor and perform expected function. The 16-bit programcounter indicates the address of the next location where the nextinstruction is to be fetched. It has 16-bit program counter input,16-bit program counter output, and some control signals likeread, write, and clear. The Program Counter (PC) contains theaddress of the instruction that will be fetched from theInstruction Memory during the next clock cycle. Normally thePC is incremented by one during each clock cycle unless abranch instruction is executed. When a branch instruction isencountered, the PC is incremented or decremented by theamount indicated by the branch offset. The instruction register(IR) has one 16-bit input and two 16-bit outputs. The instructionset describes the bit-configurations allowed in the IR .Theinstruction consists of operation codes and operand. Theprocessor support direct, register, registers indirect andimmediate addressing modes. In immediate addressing; theoperand field contains the data itself. In registers addressing, theoperand field contains the address of a data path register inwhich the data resides. In registers indirect addressing, theoperand field contains the address of a register, which in turncontains the address of a memory location in which the dataresides. In direct addressing, the operand field contains theaddress of a memory location in which the data resides. Inindirect addressing, the operand field contains the address of amemory location, which in turn contains the address of amemory location in which the data resides.

Memories are used for storage of both instructions and data. Theprocess of storing data into memory is called writing andretrieving data or operation codes from the memory is calledreading. For reading and writing data, the particular memorylocation is identified and then reading or writing is done. Thereare 64 number of 16-bit memory locations. It has 8-bit addressand 16-bit word.

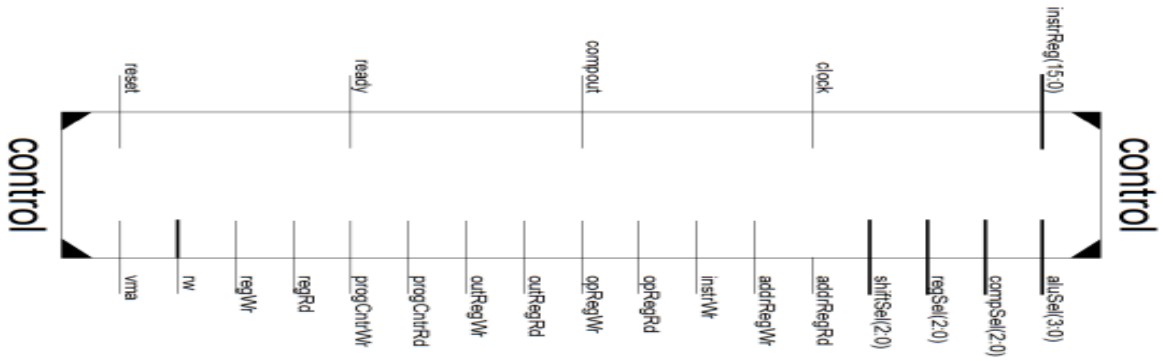

Fig. 8RTL view of Control Unit

IX. PROCESSOR WORKING

Memory module of processor contains the instructions anddata for the CPU to execute. Control Unit provide the different control signal to different processor blocks to execute the instruction saved in memory. On asynchronous reset input processor enters reset state. During this state it loads program counter with "0000" address. Instruction fetched from memory with address "0000" is loaded into instruction register. In the execute state, instruction register is decoded and based on the instruction it is moved to corresponding instruction state. After execution of each instruction, it increments program counter and fetches the next instruction and in execution, it decodes the respective instruction and moves to corresponding instruction state. Processor follows such execution cycle uptillyhe last instruction.

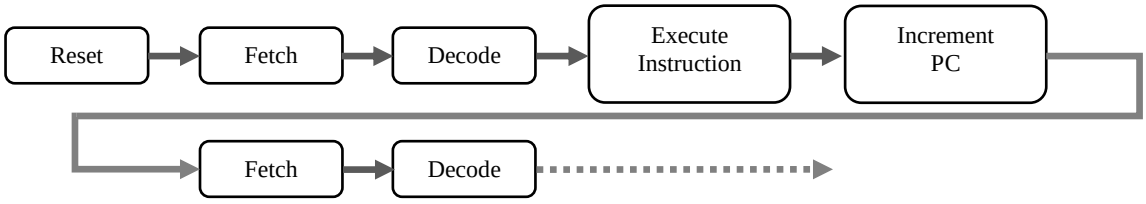

Fig. 9Instruction Execution Cycle

X. CONCLUSION

The VHDL is a versatile language which has great flexibilityof designing component and FPGA embedded processor hasthe power and ability to provide previously unachievableflexibility and performance. One can design hardware in aVHDL for FPGA implementation and to produce the RTL schematic of the desired circuit such as Xilinx ISE.After that the generated schematic can be verified usingsimulation software Active HDL / ModelSim which shows the waveforms of inputs andoutputs of the circuit after generating the appropriate testbench. We learned how to produce different arithmeticoperations and logical functions by using various selectsignals for a single circuit. Rapid implementation of parallelstructures based on FPGAs using VHDL proves to be a veryefficient, cost- effective and attractive methodology fordesign verification.Verification of the designed static elements using Xilinx ISEtool is implemented using Very High Speed HardwareDescriptive Language and Xilinx Spartan 6 FieldProgrammable Gate Array.

The 16-bit processor was describedusing VHDL and was implemented on Xilinx *XC6SLX16-2CGS324* FPGA device. The modular, hierarchicaldesign method adopted makes it easy to modify. Thedesign of ALU was extended so that it can also perform multiplication and division operation along with other arithmetic and logical operations.The 16-bit microprocessor hasa regular instruction format and flexible addressingmode.

We have compared thesimulated output results with the expected results. Fromsynthesis report, the minimum clock period that can be achievedin this proposed architecture is 52.272ns (19.131MHz). Finally, the performance of theprocessor is tested for all arithmetic and logical operation along with multiplication and division operation and thefunctionality is found correct.Further enhancements can bemade on this system by adding more number of inputs withincreased number of bit size.

REFERENCES

[1] Xiao Tiejun and Liu Fang, “16 Bit teaching Microprocessor Design and Application”, IEEE, Bursa, 2010.
[2] Oztunk E. and Sedef H., “Design of 16 Bit Microprocessor by using FPGA”, IEEE, Xiamen, 2008.
[3] V. Khorasani, B. V. Vahdat, and M. Mortazavi, “Design and implementation of floating point ALU on a FPGA processor”, IEEE International Conference on Computing, Electronics and ElectricalTechnologies (ICCEET), pp. 772-776, 2012.
[4] SuchitaKamble, Prof .N. N. Mhala, “VHDL Implementation of 8-Bit ALU”, IOSR Journal of Electronics and Communication Engineering (IOSRJECE), ISSN : 2278-2834 Volume 1, Issue 1 (May-June 2012), PP 07-1.
[5] Prof. S. Kaliamurthy& Ms. U. Sownmiya, "VHDL design of arithmetic processor", Global Journals Inc. (U.S.A), November 2011.
[6] ZainalabedinNavabi, “VHDL:Modular Design and Synthesis of Cores and Systems”, McGraw-Hill Professional; 3 edition (1 May 2007)
[7] J. Bhaskar, “VHDL Primer”, Pearson Education, 3rd edition, 2000.
[8] http://www.xilinx.com/support/documentation/data_sheets/ds160.pdf

