International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2014): 5.611

8 Bit Instructional Processor Performing ALU Operations using FPGA

Nikita V. Nandanwar[1], A. B. Diggikar[2]

[1, 2] S.P.W Engineering College, Aurangabad, India

Abstract: ***Processor is the circuitry that carries out instructions of program by performing the basic arithmetic, logical, control and transfer operations specified by the instructions and the program is set of instructions executed in proper sequence. This main purpose of this paper is to realize such program using instructions from different groups such as data transfer group, arithmetic and logical group. The main aim of this work is to check whether different instructions from different group perform functionally correct when they are to be coded successively. Code is programmed using VHDL programming. Xilinx ISE 12.4 is used and results are verified using Isim simulator. Regarding hardware results are verified using Spartan-6 XC6LX16-CS324 board along with additional inputs and outputs by interfacing it with laptop.***

Keywords: ALU, FPGA, Processor, program, Spartan 6.

1. Introduction

In general program is plan of action at accomplishing a particular objective, with details on what work is to be done, by whom, when and what means or resources will be used for the same. In computing it is a set of coded instructions that a processor can understand to solve a problem or produce a desired result.

Processor is the logic circuitry that responds to and processes instructions, whereas FPGA contains an array of programmable logic blocks which can be configured to perform various functions. This paper includes implementation of program of Microprocessor using Spartan-6 FPGA. Instructions of microprocessor from data transfer group, arithmetic group, logical group and machine control programmed are used to build a program and was then executed. Xilinx ISE 12.4 is used to program. Performance of codes is verified using Isim simulator. After which they are tested on Spartan-6 FPGA board with additional inputs and outputs.

2. Existing Work

Many Research work have been done on processor using VLSI. It includes 8 bit, 16 bit and 32 bit processor. Some of them were using VHDL while others were using Verilog. FPGA of different families were used. Recently work on 8 bit Instructional processor using Spartan-6 was done by me. It included realization of functions of 8 bit processor using Spartan 6 and Opcodes were optimized in such a manner to reduce its complexity. Functions of processor in different modules were programmed using VHDL programming. Xilinx ISE 12.4 was used and results were verified using Isim simulator. Regarding hardware results were verified using Spartan-6 XC6LX16-CS324 board along with additional inputs and outputs by interfacing it with laptop. Instructions 8085 processor was taken as reference. Instructions from Arithmetic, logical, data transfer, machine control and branching group were tested [1]. This paper explains implementation of the same in the form of program with reference to the existing work.

3. Proposed work

This paper explains a simple application of ALU, based on instructional processor using FPGA. Initially instructions were programmed and tested experimentally as separate modules. All types of instructions were programmed using Xilinx ISE 12.4 and were tested using Spartan-6 with additional set of inputs and outputs. In this paper program performing different ALU operations is coded.

Similar to conventional 8 bit processor, set of various instructions can be coded to form a single program using Xilinx and can be practically tested. Following is the code for representing ALU operations in one single program.

3.1 Program for ALU operation:

```
entity REG_A is
Port ( clk : in  STD_LOGIC;
PC: inout STD_LOGIC_VECTOR (7 downto 0);
opcode: in  STD_LOGIC_VECTOR (7 downto 0);
data1: inout  STD_LOGIC_VECTOR (7 downto 0);
data2 : inout  STD_LOGIC_VECTOR (7 downto 0);
A : inout  STD_LOGIC_VECTOR (7 downto 0);
B : inout  STD_LOGIC_VECTOR (7 downto 0);
C : inout  STD_LOGIC_VECTOR (7 downto 0);
hlt : out  STD_LOGIC;
Z : inout  STD_LOGIC ;
cy :inout  STD_LOGIC );
end REG_A;
architecture Behavioral of REG_A is
signal temp : STD_LOGIC_VECTOR (7 downto 0);
signal X : STD_LOGIC_VECTOR (8 downto 0);
signal EB : STD_LOGIC ;
begin
process (clk)
variable count : integer range 0 to 6 ;
begin
data1<= "00011000";
```

```
data2<="00001111";
B<= "10000000";
if (clk='1') then
   if (count=0 and opcode="00101000") then ----MVI A,data
      Pc<= Pc+1;
      A<=data1 ;
   end if ;
   if (count=1  and opcode="00001000") then  ---- MOV B,A
      Pc<= Pc+1;
      B<=A ;
   end if ;
   if (count=2 and opcode="01000001") then  ----- ADD B
      pc <= pc + 1 ;
      A<= A + B ;
      if ( A ="00000000") then
      Z<='1' ;
      else
         Z<='0' ;
      end if ;
      if (  A > "11111111")then
      cy <= '1' ;
      else
      cy<= '0' ;
      end if ;
   end if ;
if (count=3  and opcode="00101010") then  -----MVI C, data
   pc <= pc + 1 ;
   C <= data2 ;
end if ;
if (count=4  and opcode="10000010") then   ----- ANA C
   pc <= pc + 1 ;
   A<= A and C;
      if ( A ="00000000") then
      Z<='1' ;
      else
      Z<='0' ;
      end if ;
      if (  A > "11111111")then
      cy <= '1' ;
      else
      cy<= '0' ;
      end if ;
   end if ;
   if (count=5  and opcode="01100000") then  ------ INR A
      pc <= pc + 1 ;
      A<= A + 1 ;
         if ( A ="00000000") then
         Z<='1' ;
         else
         Z<='0' ;
         end if ;
         if (  A > "11111111")then
         cy <= '1' ;
         else
         cy<= '0' ;
         end if ;
   end if ;
   if (count=6  and opcode="11010000") then  ------- HLT
      pc <= pc + 1 ;
      hlt <= '1';
      A<= "00000000";
      B<= "00000000";
      C<= "00000000";
      Z<='0';
      cy<='0';
      data1<= "00000000" ;
      data2<= "00000000" ;
   end if ;
count := count + 1;
end if ;
end process ;
end Behavioral;
```

3.2 Program Description

In this program,
Clk: It indicates input clock pulse. It is 25ns.
PC: It refers to program counter. It is initially reset.
Opcode: It is Operational code. It differs with instruction.
Data1: It is first immediate data to be moved.
Data2: It is second immediate data to be moved.
A: It represents register A.
B: It represents register B.
C: It represents register C.
Hlt: It is signal which represents Halt condition.
Z: It indicates Zero flag.
Cy: It indicates carry flag.

This program is to perform some ALU operations. Initially value of data1 and data2 are initialized which are then to be moved to registers. In this program seven instructions are performed in successive manner. Starting with MVI A, data. i.e. immediate data is taken in register A for the Opcode "00101000". Then the contents of register A are copied in register B using MOV B, A for the Opcode "00001000". Then contents of register A and register B are added using ADD B instruction for the Opcode "01000001". Further immediate data is taken in register C using MVI C, data for the Opcode "00101010". Then the contents of register C and register A are logically ANDed using ANA C instructions for the Opcode "10000010". Further the content of register A is incremented by one using INR A instruction for the Opcode "01100000". Finally the program ends with hlt instruction. This will reset all registers along with program counter and Flags.

3.3 Experimental Result

Above program is simulated using Isim simulator. All the instructions used in the program performed functionally correct. Simulation and RTL schematic of program is as follow:

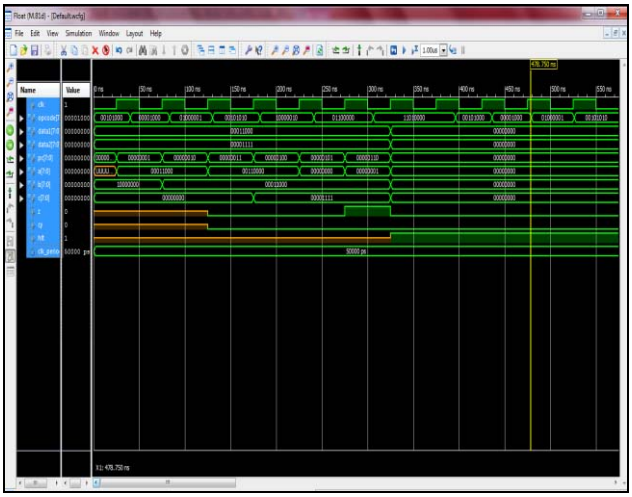

Figure 1: Simulation of Program

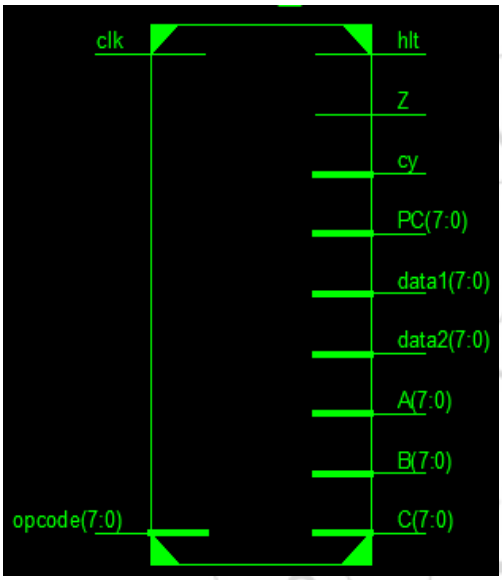

Figure 2: RTL Schematic of program

4. Experimental Setup

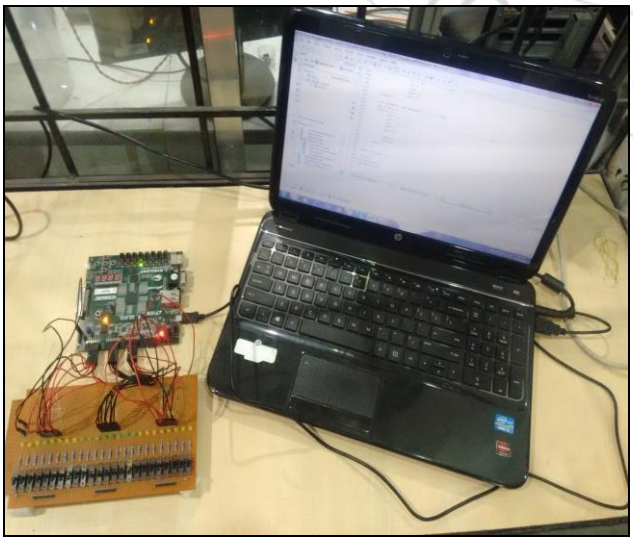

Figure 3: Experimental setup

The program was also tested practically. Experimental setup include Spartan-6 FPGA (XC6LX16-CS324) board, additional switches and LEDs as inputs and outputs respectively, laptop with Xilinx ISE 12.4 and USB cable for interfacing FPGA board with laptop. 8 Bit Opcode was given as input through switches on Spartan-6 board and the 8 bit contents of register A, register B and register C is read using LED's of additional board. The experimental setup required for testing the program is shown in figure 3.

5. Conclusion and Future Scope

In this way simple program using instructions from different groups such as data transfer group, arithmetic and logical group was programed. The main aim of this work was to check whether different instructions from different group were working functionally correct when they are to be coded successively. The entire model was synthesized using Xilinx ISE 12.4. Isim simulator was used for simulation. Following the simulation code was dumped on to a Spartan-6 FPGA board for functional verification, which was carried out successfully. It was found that both simulation and practical testing on board gave expected results. Future scope includes consideration of more bits i.e. 16, 32, etc. and also to increase number of operations performed by processor.

References

[1] Nikita V. Nandanwar, P. R. Thorat , "8 Bit Instructional Processor Using FPGA", International Journal of Science and Research (IJSR), Volume 4 Issue 11, November 2015

Author Profile

Nikita V. Nandanwar received B.E degree in Electronics and Telecommunication Engineering from M.S.S College of Engineering and Technology, Jalna in June 2012. During 2013-2014, she worked as Lecturer in Electronics department at polytechnic institute, Aurangabad. She is now completing her M.E in Electronics at S.P.W Engineering College, Aurangabad.