International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 3, March - 2013

4-Bit Arithmetic And Logic Unit Design Using Structural Modelling In VHDL

Rupali Jarwal[1] and Ulka Khire[2]
1, 2 Microelectronics and VLSI Design,
Electronics & Instrumentation Engineering department, SGSITS, Indore, M.P., India

Abstract

This paper presents design concept of 4-bit arithmetic and logic unit (ALU). Design methodology has been changing from schematic design to HDL based design.

We proposed arithmetic and logic unit using VHDL structural and dataflow level design. Each module of ALU is divided into smaller modules. All the modules in arithmetic and logical unit design are realized using VHDL design. Functionalities are validated through synthesis and simulation process. Besides verifying the outputs' ,the outputs' timing diagram and interfacing signal are also tracked to ensure that they adhere to the design specification.ALU using VHDL fulfils the needs for different high performance application.

Index terms-Arithmetic and logic unit (ALU), Very high scale integrated circuit Hardware Description Language.

1. Introduction

The ALU, or the arithmetic and logic unit, is the section of the processor that is involved with executing operations of an arithmetic or logical nature. In ECL, TTL and CMOS, there are available integrated packages which are referred to as arithmetic and logic units (ALU). The logic circuitry in this units is entirely combinational (i.e. consists of gates with no feedback and no flip-flops).The ALU is an extremely versatile and useful device since, it makes available, in single package, facility for performing many different logical and arithmetic operations.

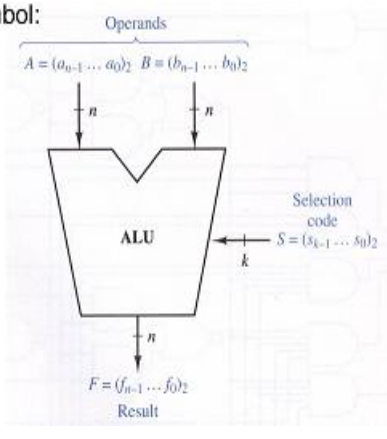
Arithmetic Logic Unit (ALU) is a critical component of a microprocessor and is the core component of central processing unit. ALU can perform all the 16 possible logic operations or 16 different arithmetic operations on active HIGH or active LOW operands. . Arithmetic instructions include addition, subtraction, and shifting operations, while logic instructions include Boolean comparisons, such as AND, OR, XOR, and NOT operations.

2. Block diagram description

ALU works in conjunction with the register array for many of these, in particular, the accumulator and flag registers. The accumulator holds the results of operations, while the flag register contains a number of individual bits that are used to store information about the last operation carried out by the ALU. More on these registers can be found in the register array section.

Example of modular design: ALU

• ALU logic symbol:



Arithmetic and logic unit consists of two blocks for different operations-
a. Arithmetic operations.
b. Logical operations.

Addition and subtraction

These two tasks are performed by constructs of logic gates, such as half adders and full adders. While they may be termed 'adders', with the aid of they can also perform subtraction via use of inverters and 'two's complement' arithmetic.

A binary adder-subtractor is a combinational circuit that performs the arithmetic operations of addition and subtraction with binary numbers. Connecting n full adders in cascade produces a binary adder for two n-bit numbers.

Logical operations

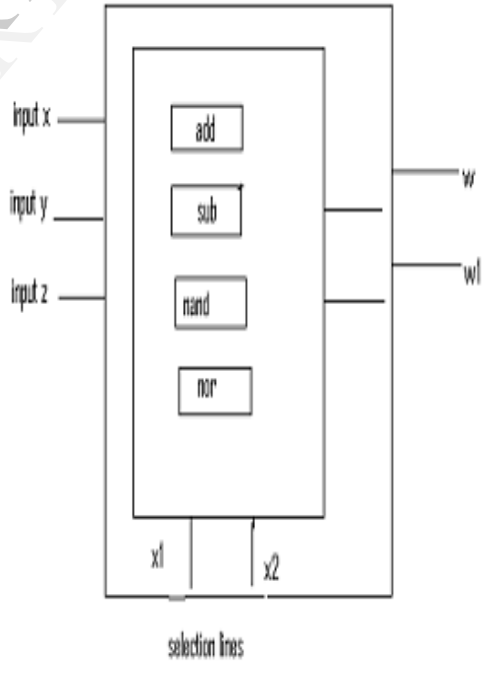
Further logic gate s are used within the ALU to perform a number of different logical tests, including seeing if an operation produces a result of zero. Most of these logical tests are used to then change the values stored in the flag register, so that they may be checked later by separate operations or instructions. Others produce a result which is then stored, and used later in further processing.

2. Proposed ALU using VHDL code

```
architecture Behavioral of alu is
signal p,q,r, s,t,u,v:STD_LOGIC;
Component nand1 is
   Port (a: in STD_LOGIC;
         b: in  STD_LOGIC;
         c : out  STD_LOGIC);
end component;
component nor1 is
   Port ( a : in  STD_LOGIC;
         b : in  STD_LOGIC;
         c : out  STD_LOGIC);
end component;
component ader1 is
   Port ( a : in  STD_LOGIC;
         b : in  STD_LOGIC;
         c : in  STD_LOGIC;
 d : out  STD_LOGIC;
 e : out  STD_LOGIC);
end component;
component asub1 is
Port ( a : in  STD_LOGIC;
         b : in  STD_LOGIC;
         c : in  STD_LOGIC;
         d : out  STD_LOGIC;
         e : out  STD_LOGIC);
end component;
begin
a1:nor1 port map (x,y,q);
a2: nand1 port map(x,y,q);
a3: ader1 port map(x,y,z,r,s);
a4:asub1 port map (x,y,z,t,u);
process (p,q,r,s,t,u,v,x1,x2)
begin
w<=((not x1)and (not x2)and r)or
((not x1)and x2 and t)or
(x1 and (not x2)and p)or
(x1 and  x2 and q);
w1<=(not x2 and s )or (x2 and u);
end process;


end Behavioral;
```

3. Implementation



Block diagram of ALU

4. Simulation Results

Design is verified through simulation, which is done in a bottom-up fashion. Small modules are simulated in separate testbenches before they are integrated and tested as a whole. All four arithmetic operations available in the design are tested with the same inputs. The sequence of operations done in the simulation is addition. The results of operation on the test vectors are manually computed and are referred to as expected result.

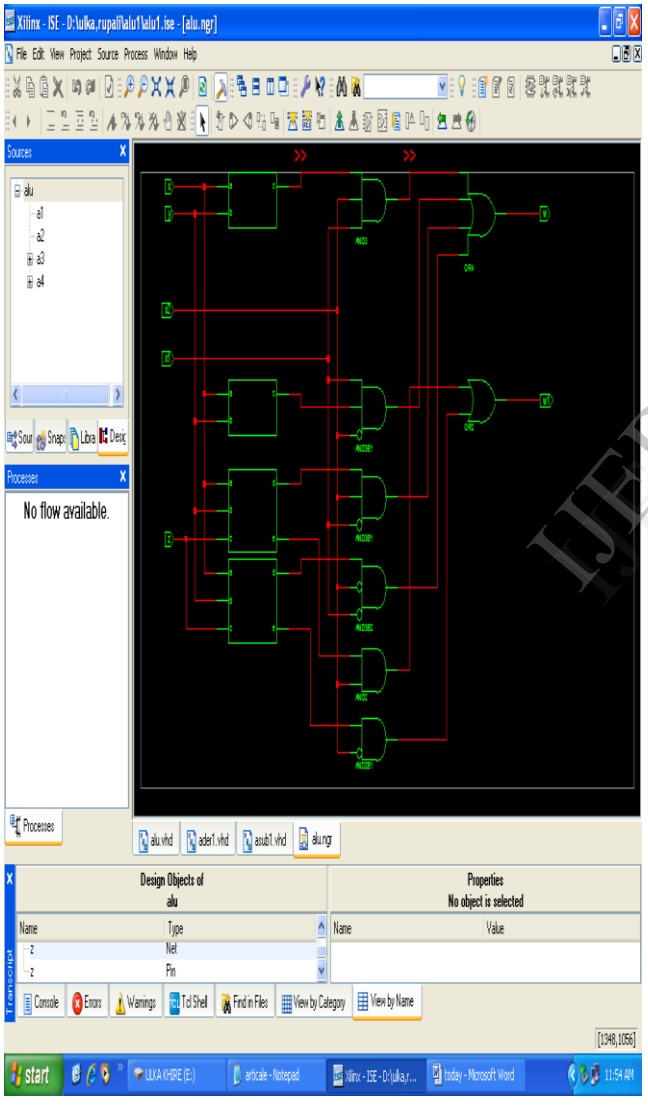


Fig 2.RTL view of 4 bit ALU

Simulated Waveform

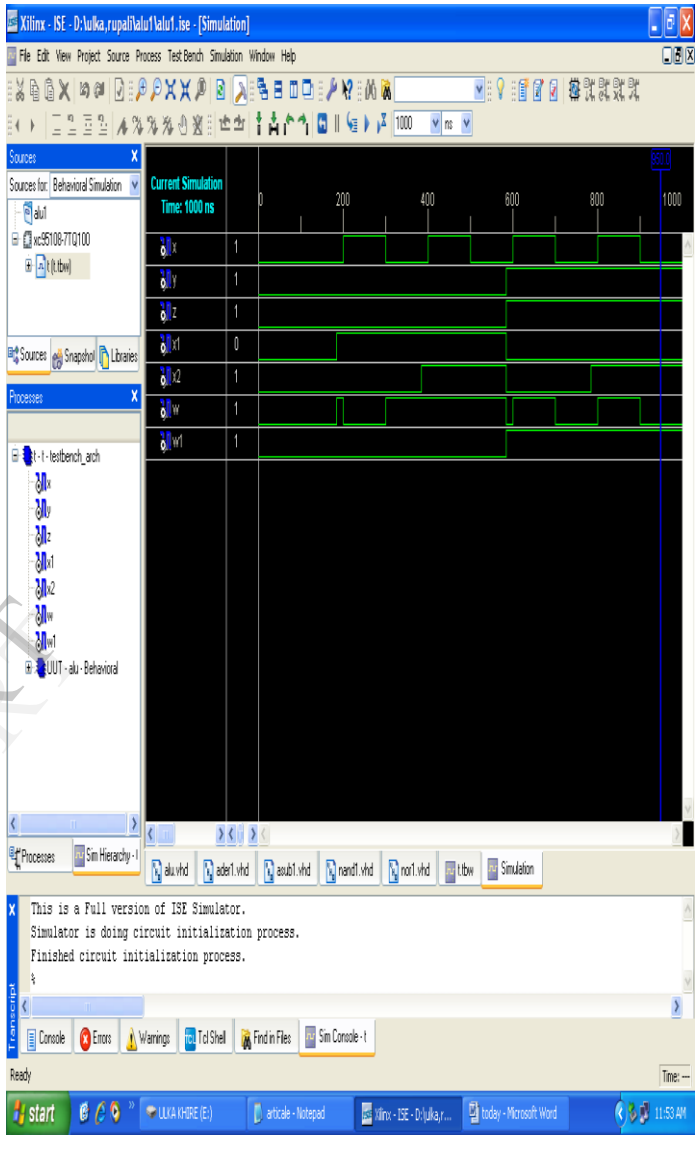


Fig 3. Output waveform

5. Conclusion

In this paper, we have proposed efficient VHDL behavioural coding verification method. We have also proposed several algorithms using different design levels.

Our proposals have been implemented in Verilog and verified using Xilinx ISE 10.1 analyzer. We have reduced the number of bus lines and all the designs have been implemented and tested. This ALU design using VHDL is successfully designed, implemented, and tested. Currently, we are conducting further research that considers the further reductions in the hardware complexity in terms of synthesis and finally download the code into Altera SPARTEN-3E: FPGA chip on LC84 package for hardware realization.

6. Acknowledgement

We gratefully acknowledge the Almighty GOD who gave us strength and health to successfully complete this venture. We wish to thank lecturers of our college for their helpful discussions. We also thank the other members of the VHDL synthesis group for their support.

7. Reference

1. Brown S., Vranesic Z "Fundamental of Digital Logic Design with VHDL" McGraw Hill, 2nd Edition.
2. Bhasker J, "*A VHDL Primer*", P T R Prentice Hall, Pages 1-2, 4-13, 28-30.
3. Jiang Hao, Li Zheying, "FPGA design flow based on a variety of EDA tools" in *Micro-computer information*, 2007(23)11-2:201-203.
4. Xilinx, Inc. Xilinx Libraries Guide, 2011.
5. Sewak K, Rajput P, Panda Amit K, "FPGA Implementation of 16 bit BBS and LFSR PN Sequence Generator:
6. *A Comparative Study*", In Proce. of the *IEEE Student Conference on Electrical, Electronics and Computer Sciences 2012*, 1-2 Mar 2012, NIT Bhopal, India.
7. Douglas L. Perry, ***VHDL,*** McGraw-Hill, **1995.**
8. "Digital Integrated Circuits" by Jan M. Rabaey , Anantha Chandrakasan and Borivoje Nikolic