ISSN (Online) 2321-2004
ISSN (Print) 2321-5526

INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH IN ELECTRICAL, ELECTRONICS, INSTRUMENTATION AND CONTROL ENGINEERING
Vol. 3, Issue 2, February 2015

Verification & Design Techniques Used in a Graduate Level VHDL Course

Prof. Swati Agrawal, BE, MS (SUNY, Buffalo, NY USA)[1]

Associate Professor, Department of Electronics & Telecommunication, Bhilai Institute of Technology, Durg[1]

Abstract: The use of the VHSIC Hardware Description Language (VHDL) has become very important to the simulation and implementation of digital systems in both industry and educational settings. Al-though VHDL is a powerful language with many capabilities, it has downfalls when considering the difficulty in learning the language as well as its limited capabilities for transitioning a design from initial concept to design entry and verification stages. This paper discusses techniques used to teach the VHDL design methodology to graduate students, as well as methods used to go through a complete design cycle from initial concept to final implementation. VHDL design techniques were developed using various projects and homework assignments, and different approaches to implementing the same function allowed direct comparisons of the speed and size of the designs. Different processes for taking a design from ini-tial concept through chip implementation were discussed, and one example of the process is discussed here. This paper describes the implementation of full adder using VHDL technology which meets less complexity requirement ,it also shows how efficiently digital system ie Full Adder is implemented upto layout level results shows technological map RTL view ,chip floor plan ,chip layout ,output waveforms showing voltage Vs time relations and verification of truth table .

Keywords: VHDL, RTL view, HDL

INTRODUCTION

The use of Hardware Description Languages (HDL's) has been steadily increasing as digital designs become larger and more complex. Previously used methods such as schematic capture have not been as well suited to design re-use and rapid prototyping of large chip designs. Although not new, HDL's have recently become popular with the widespread use of two similar but distinct languages. The older of these languages, VHDL (VHSIC Hardware Description Language)[1], is heavily used by U.S. defense agencies and large corporations, as well as a majority of European companies. VHDL was first adopted as language standard in 1987, with a major revision occurring in 1993. The other major HDL currently in use is Verilog[2], which is more common among smaller companies and corporations with fewer ties to the military. While both languages are very powerful and can be used to obtain the same results, the VHDL syntax is more complex than Verilog. This paper describes the learning methods used in teaching an upper level course on VHDL. Students are expected to use their new knowledge to implement designs in VHDL.

DESIGN CYCLE IMPLEMENTATION

One of the key objectives of this course is for students to learn to implement an actual design in hardware by starting with the initial design concept. This extends the content of the course beyond VHDL programming, but instead focuses to applying VHDL to real design issues. Certain steps are traditionally carried out in a design, but the primary ones we focused on are:

1. Define requirements and specifications.
2. Develop a system model for the design concept which meets the specifications.
3. Implement the system model in behavioral VHDL.
4. Simulate the behavioral VHDL to determine if it meets the system specifications.
5. Synthesize the design using the behavioral VHDL description.
6. Verify through simulation that the synthesized design still meets the requirements.
7. Implement (program) the synthesized design into hardware.
8. Test the actual device and circuit, and compare the performance to original requirements.

As with any design, the first step is to define the requirements and specifications. The students learn there is no magi-cal method for achieving this, but rather it may take consider-able research, brainstorming, and some intuition based on prior experience. These requirements are reviewed throughout the design cycle, and it is not uncommon for changes to the specifications to occur as development progresses.

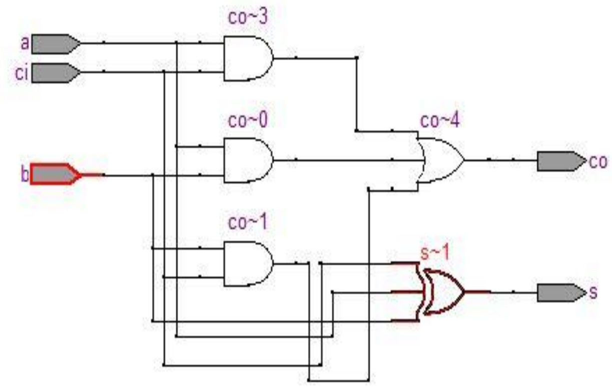
The next step is to develop a system model for the design and verify the model meets the design requirements. Implementation of the system model may occur in various forms. One of the options is to use the modeling aspects of VHDL to write and simulate a VHDL model of the design. The VHDL code does not have to be synthesizable at this point to verify the system model.

Once the system model is verified the synthesizable VHDL must be created based on the model. For this course the creation of the VHDL is strictly performed by hand. In industry that is not always the case. As with any similar capability, it is important to understand how the VHDL is created and the integrity of the design should be verified.

A VHDL simulator may now be used to verify the VHDL description still behaves as expected. If so, a synthesis tool may be used to map the VHDL into logic. If the behavior is not correct, then the VHDL must be modified before synthesis. Once the synthesis tool has successfully

DOI 10.17148/IJIREEICE.2015.3201

mapped the VHDL into logic, the logic design (structural VHDL) may now be simulated with the VHDL simulator to verify the design still meets the system requirements. If the design fails any requirement, then the designer needs to return to one of previous steps in an attempt to fix the problems(s). A synthesized design which has been validated through simulation may now be implemented on the physical device(s). Real physical tests and analyses are now required as the final step in the design cycle. If the resulting circuit does not meet the appropriate requirements then the revisions are typically required at one of the previous steps.



Logic diagram of Full Adder

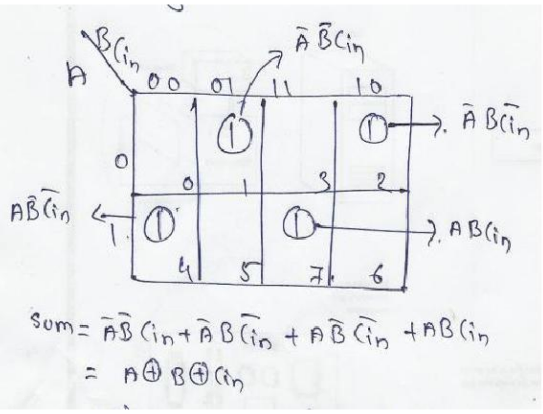
From the truth table, a circuit will produce the correct sum and carry bits in response to every possible combination of **A, B**, Cin

FULL ADDER DESIGN

Full adder is a combinational circuit that has a ability to add two bits and a carry input and produces sum bit and carry bit as output .full adder adds two bits A and B and carry from previous column called as carry input. fig (1) shows logic diagram of full adder table (1)shows truth table of full adder

Inputs			**Sum**	**Carry**
A	B	C	S	Cout
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Now using K-Map for sum column & Carry Column we get the following equations



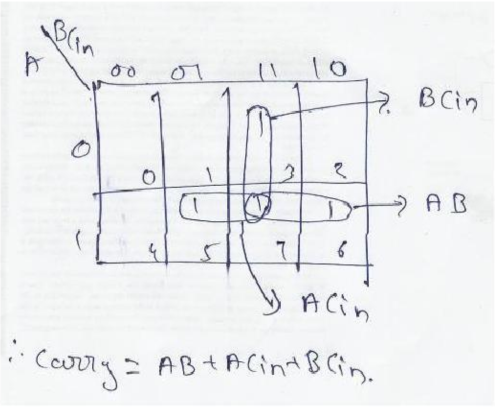


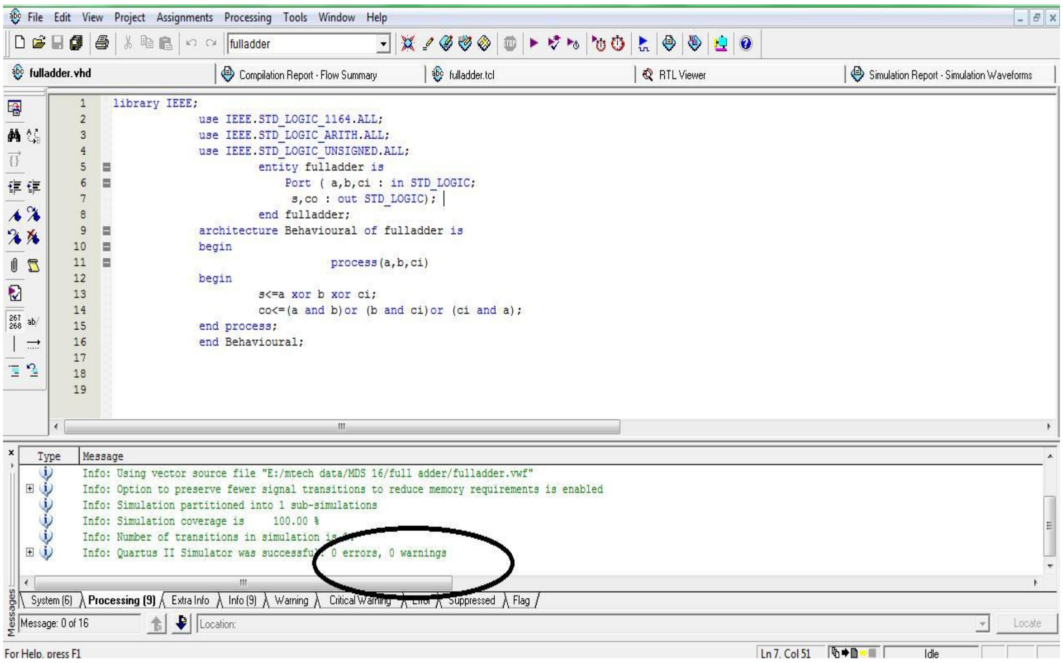
Fig1: Block Diagram

SIMULATION RESULTS

All simulation for this full adder has been performed using Quartus II And Microwind tool. A fully integrated Full Adder has been designed, fig (2) shows VHDL code compilation of Full Adder with zero errors and zero warnings. fig(3)shows output waveform of Full adder which verifies the truth table.fig (4)shows Technological map view .fig(5)shows layout of full adder .fig(6) shows schematic of full adder in analog domain.

DOI 10.17148/IJIREEICE.2015.3201

Fig 2:VHDL Code compilation


File Edit View Project Assignments Processing Tools Window Help
fulladder
fulladder.vhd
Compilation Report - Flow Summary
fulladder.tcl
RTL Viewer
Simulation Report - Simulation Waveforms

```
library IEEE;
            use IEEE.STD_LOGIC_1164.ALL;
            use IEEE.STD_LOGIC_ARITH.ALL;
            use IEEE.STD_LOGIC_UNSIGNED.ALL;
                    entity fulladder is
                        Port ( a,b,ci : in STD_LOGIC;
                         s,co : out STD_LOGIC);
                    end fulladder;
            architecture Behavioural of fulladder is
            begin
                                 process(a,b,ci)
            begin
                    s<=a xor b xor ci;
                    co<=(a and b)or (b and ci)or (ci and a);
            end process;
            end Behavioural;
```

```
Type  Message
      Info: Using vector source file "E:/mtech data/MDS 16/full adder/fulladder.vwf"
      Info: Option to preserve fewer signal transitions to reduce memory requirements is enabled
      Info: Simulation partitioned into 1 sub-simulations
      Info: Simulation coverage is     100.00 %
      Info: Number of transitions in simulation is 0
      Info: Quartus II Simulator was successful. 0 errors, 0 warnings
```

Message: 0 of 16

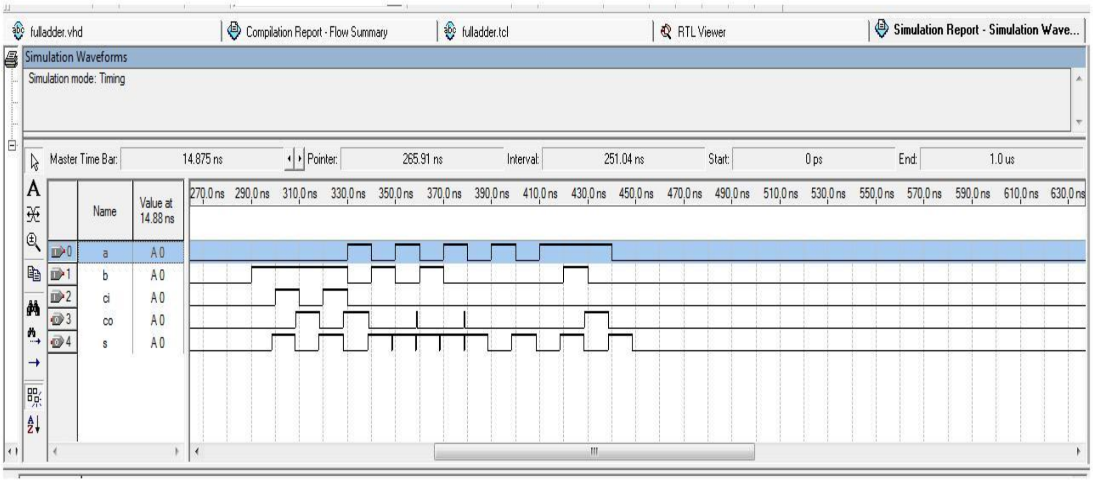
Location:

For Help, press F1

Ln 7, Col 51

Idle

Fig 3: Output Waveform



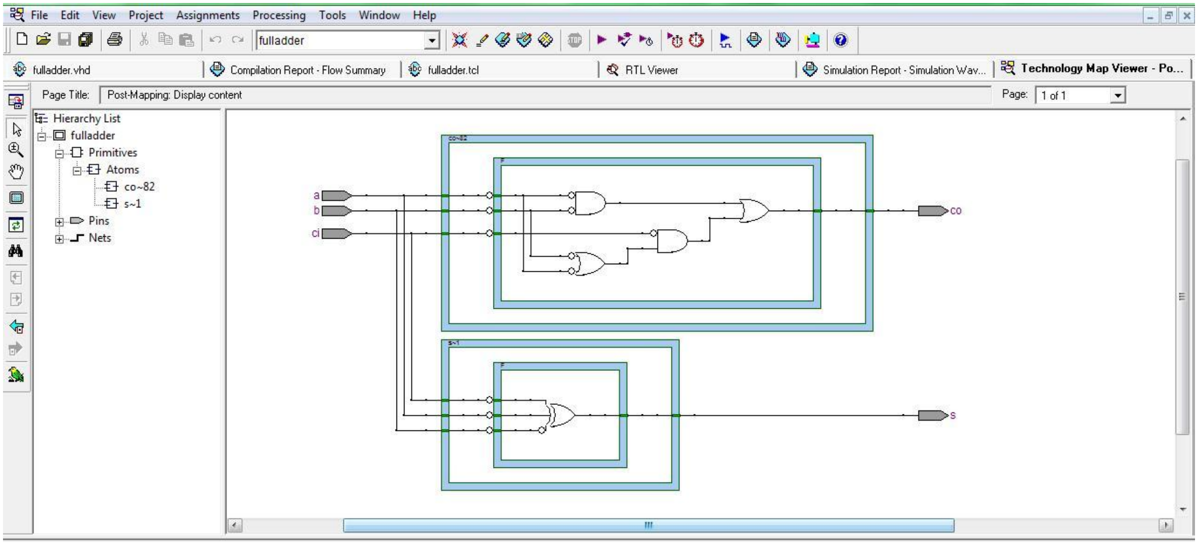
Simulation Waveforms

Simulation mode: Timing

Master Time Bar: 14.875 ns   Pointer: 265.91 ns   Interval: 251.04 ns   Start: 0 ps   End: 1.0 us

	Name	Value at 14.88 ns
0	a	A 0
1	b	A 0
2	ci	A 0
3	co	A 0
4	s	A 0

Fig 4: Technological map view


File Edit View Project Assignments Processing Tools Window Help
fulladder
fulladder.vhd
Compilation Report - Flow Summary
fulladder.tcl
RTL Viewer
Simulation Report - Simulation Wav...
Technology Map Viewer - Po...
Page Title:
Post-Mapping: Display content
Page:
1 of 1
Hierarchy List
fulladder
Primitives
Atoms
co~82
s~1
Pins
Nets
a
b
ci
co
s
co~82
s~1

DOI 10.17148/IJIREEICE.2015.3201

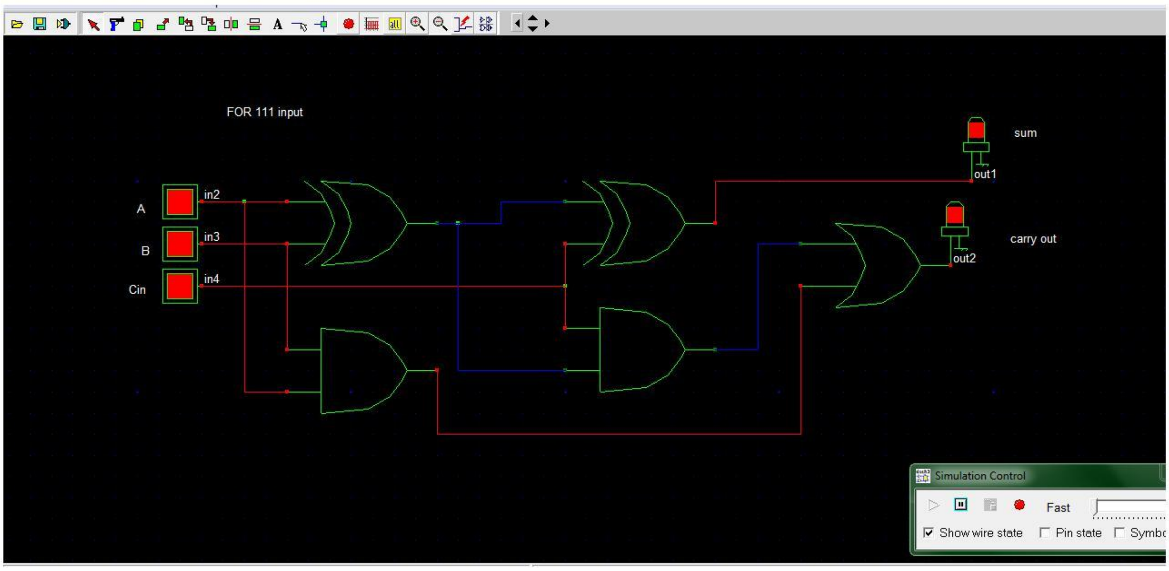


Fig 5: Schematic of full adder in analog domain

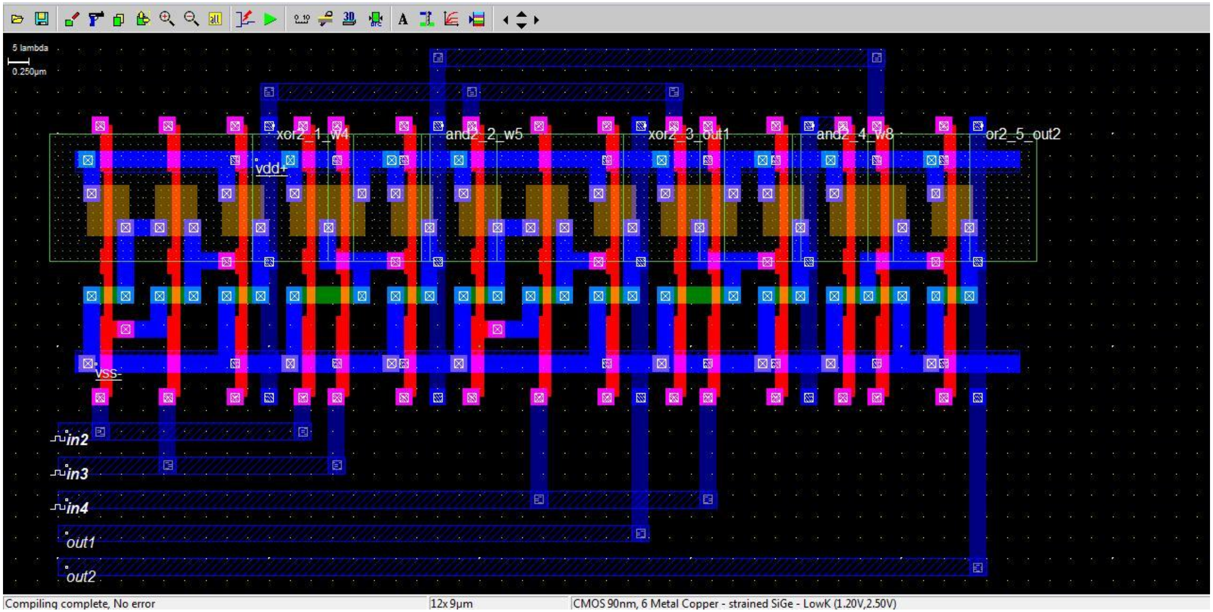


Fig 6 Layout of full Adder circuit

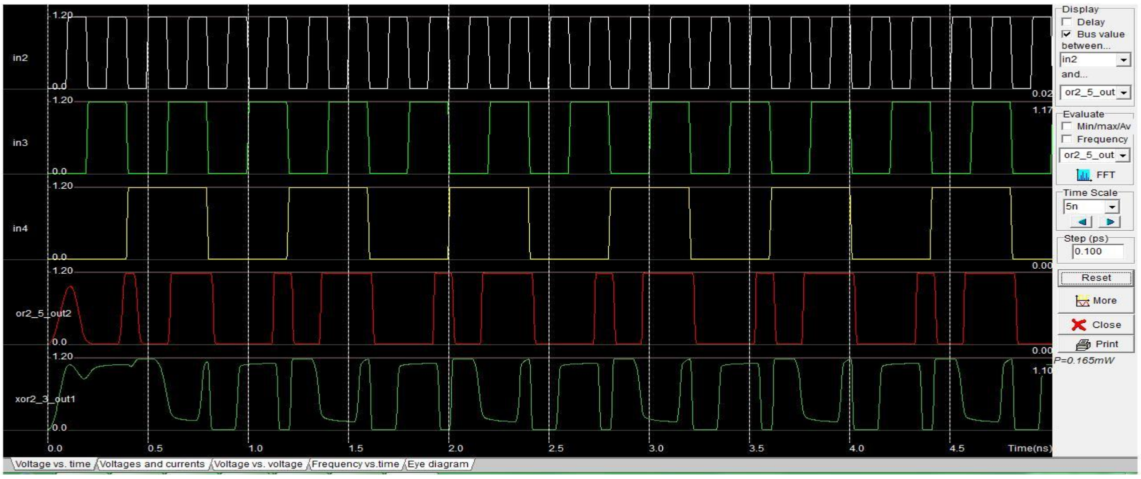


Fig 7: Voltage VS Time relationship in analog domain

DOI 10.17148/IJIREEICE.2015.3201

CONCLUSION

CMOS 90nm model is used to design a layout of full adder. Simulation results shows successful compilations of VHDL code and its conversion into verilog file which is used to make a layout using microwind tool. tools used are Quaruts II ,Dsch& microwind . From both performance standpoint and cost standpoint, these results show that CMOS is very competitive with available technologies.

REFERENCES

[1] D. Pellerin and D. Taylor, *VHDL Made Easy*, Prentice Hall, 1997.
[2] S. Palnitkar, Verilog HDL, Prentice Hall, 1996.
[3] P.Ashenden, *The designer's guide to VHDL*, Morgan Kaufmann Pub. Inc., 1996.

BOOKS

1. Stephen Brown, Fundamentals of digital logic with Verilog design (Tata McGRAW Hill)
2. Jayaram Bhasker,A VHDL Primer (PTR Prentice Hall Englewood cliffs, New Jersey 07632