JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue

JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)

An International Scholarly Open Access, Peer-reviewed, Refereed Journal

DIGITAL MODULAR IMPLEMENTATION OF DSP SYSTEMS FOR DATA AND DESIGN HIDING APPLICATIONS

Yalamanchi Teja Sri, Research Scholar, Department of ECE, International School Of Technology and Sciences (Women), Affiliated to JNTUK, NH-16, Eastgonagudem, Rajanagaram,Rajamahendravaram,AndhraPradesh,53329, India.

R. Prasad,M.Tech, Assistant Professor, Department of ECE, International School Of Technology and Sciences (Women), Affiliated to JNTUK, NH-16, Eastgonagudem, Rajanagaram, Rajamahendravaram, Andhra Pradesh 53329, India.

Abstract :

This thesis presents a novel approach to design confounded circuits for digital signal processing applications using high level transformations, a key based confounding finite-state machine (FSM), and a reconfiguration. The aim is to design Digital Signal Processing circuits which are reutilized by the specific operational methods by the designer. This design aims at the high-level transformations of repeated state graphs which have been utilized for area, speed, power compromises. It is the primary idea to develop a design flow to use high level transformations that not solely meet these tradeoffs however additionally change the architectures each structurally and functionally. Many modes of operations are introduced for obfuscation wherever the outputs are meaningful from an indication process point of view, however are functionally incorrect. Examples of such methods incorporates the next order digital filter primarily based applications that are executed in an exceedingly time multiplexed fashion. Many meaningful modes are make use to reconfigure the filter order for various applications. Still existing modes may correspond with non meaningful modes. The configure data controls varied modes of the circuit operation. Functional obfuscation is fulfill by the right value key, and configures data. Incorrect input key is unsuccessful to change the reconfigurator and an incorrect configure data generates either a meaningful however non functional or non understandable mode. Here we have a liability to perform some chance of activating the right mode, which ends up the reduced operations to an obfuscated DSP circuit. The efficiency of proposed implementation is verified with IMAGE SCALLING WITH INTERPOLATION AND DECIMATION design, strong high level obfuscation is proved and analyzed for various key sizes.

I. Introduction

The hardware security could be a severe problem that has leads to plenty of work on hardware security of piracy and intellectual property (IP). The protection of hardware is often generally classified into 2 ways that 1) Authentication based approach, 2) Obfuscation based approach. Confounding based approach is an interest thing in the present thesis, which is an approach that change an application or a design into one that isfunctionally identical to the original but is significantly more difficult to reverse engineer. The hardware protection strategies are attained by neutering the human readability of the Hardware description language (HDL) code, or by encrypting the source code base cryptographic techniques. In present days, varieties of hardware prevention schemes are grower up that modify the finite-state machine (FSM) representations to change the circuits. This thesis, for the 1st time, presents design of confounded DSP circuits via high-level transformations that are difficult to reverse engineer. Depends on photolithography, we are generating layout options of smaller dimensions, less than the wavelength of the light, which requires progressively, advanced OPC and alternative DFM techniques at increasing layout space and price.In modern days, the improved in variety of software system based protection solutions are changed to hardware based protection solutions for a lot of improved resistance to software system based security problems. The systems range from smartcards to specialized secure co-processing boxes, where as hardware provides the source of security and trust for a number of security primitives. However, in recent years, it's been brought into light that hardware is additionally subject to variety of security threats. The current technologies largely specialized in information leak from a hardware system. A chanllenger could get crypto graphical keys and secured information from a system by testing reverse engineering, or side- channel analysis.In present international IC trade, a offer chain opponent, like an IP provider, an IC design house, a CAD company, or a metal works could have access to the source code of the design, and should simply destruction a hardware system by planting time bombs that compromise hardware computation integrity, or making back doors that change data leak, by passing access control mechanisms at higher levels. The recently-released Comprehensive National Cyber Security Initiative has identified this supply chain risk management drawback as a high national priority a supply chain adversary‟s capability is frozen in his information on the hardware design.

II. LITERATURE REVIEW

Integrated circuit (IC) technology is the empower technology for a whole host of creative devices and systems that have changed the way we live. For the invention of the integrated circuit the Nobel Prize in Physics is received by Jack Kirby and Robert Nonce. If we are not with the integrated circuit, neither transistors nor computers would be as important as they are today. VLSI systems are small in size and consume less power which leads to less area of occupation than the discrete components used to develop electronic systems in the early 1960s.Many more transistors are accustomed construct systems in

integration method, which allows a lot of evaluating power to be applied for finding a retardant. Integrated circuits are abundant easier than discrete element systems once

scrutiny in design and manufacture and are a lot of reliable that produces it potential to develop special purpose systems that are a lot of systematic than all-purpose computers for the work hand.In previous paper with the title Protecting DSP circuits through Obfuscation theauthor implemented the confounded code for filters like IIR, FIR with 1st order, 2nd order and 3rd order filters. In this proposed thesis confoundig is used for DSP circuits like digital image with image scaling algorithm

Field Programmable Gate Arrays (FPGA):

The multi-level logic functions are implemented in a block of programmable logic which is known as FPGA (Field Programmable Gate Arrays). FPGAs implement large functions by employ separate commodity chips by programming, that can be utilized in present days. However, tiny blocks of FPGA logic are useful elements on- chip to permit the user of the chip to customize a part of the chip‟s logical function. The combinatory logic functions which implement on FPGA block and interconnectto be ready to construct multi-level logic functions. In FPGA‟s we have the different programming technologies, but most logic processes are different form to implement anti fuses or similar hard programming technologies, so we will concentrate on programmed Static Random Access Memory programmed FPGAs.

III. BLOCK DIAGRAM AND DESCRIPTION :

This chapter gives the information about the block diagram and description of each block with program code execution. Each block program execution is described with the flow charts. The high level transformations used here are speedup transformation and power transformation. The proposed design steps are described.

IV.DESIGN SOFTWARES AND LANGUAGE

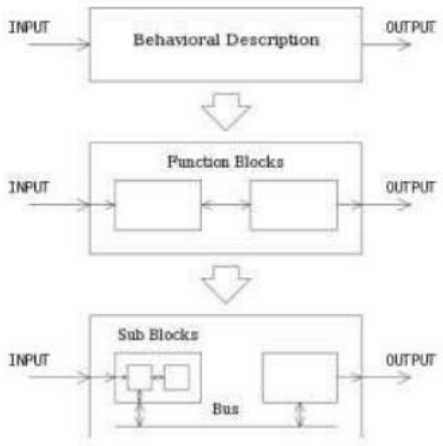
In this chapter I mentioned about the software used and coding language. For the proposed design HDL designer software is used and to know the simulation result Modelsim software is utilized. In VLSI technology design we use different languages like VHDL, Verilog, System verilog etc. For this design I used Verilog language. In this the creation of a project in HDL designer, how to simulate the design implementation and about the verilog language with some examples are explained.The benefits of hdl designer is to bring tools for the execution of designs using VHDL. The tools we'll use are Modelsim from Mentor Graphics for VHDL simulation and HDL designer that additionally return from Mentor Graphics for design and architectural/graphical design approach.

HDL Designer:

We 1st outline the fundamentals of the hdl designer atmosphere and element ideas. To demonstrate the chance of hierarchic design we have a tendency to utilize the flexibility to form one or a lot of views to every element. We have a tendency to 1st add the component symbol and a behavioural design. The design is then simulated and valid.

Top-down design:

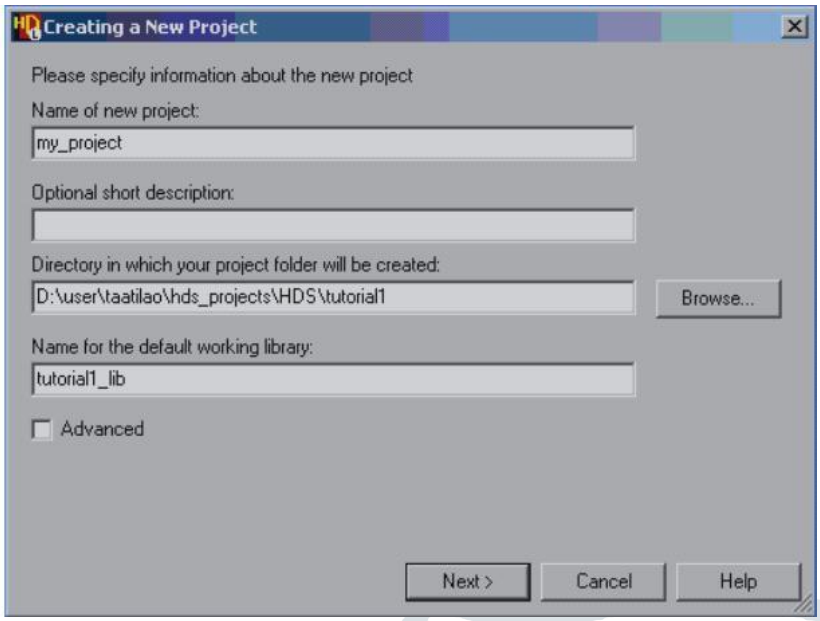
Here the overview of a model as a step within the design flows of sequential clarification. To every model variety of sub blocks exist, that are achieved as sufficiently outlined blocks are obtained. The objectives for every model are declared before in design. In top-down design we tend to develop the system stepwise by synthesizing and confirming every level. The design levels are in turn divide into sub blocks. This method for decomposition is continuous till sufficiently small blocks are obtained, as shown in Figure 4.1. The process of putrefaction with ordered clarification additionally guarantees that larger and much of necessary problems are rectified before the flowery problems.The process of putrefaction with ordered clarification additionally guarantees thatlarger and much of necessary problems are rectified before the flowery problems. By simply making little changes between ordered models, the managing and validation of models becomes easier

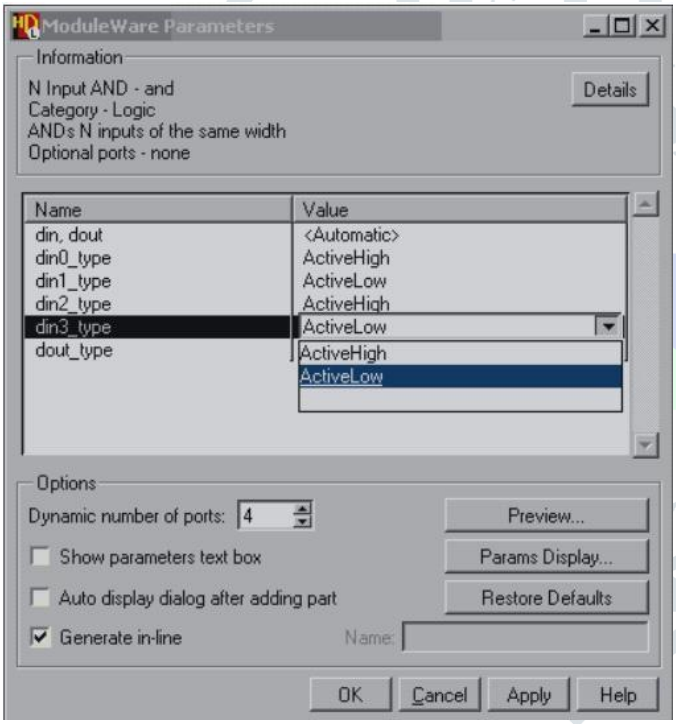


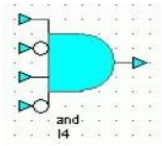
Since the issue that modifies the structure of every level withinthe design method, the design will at all levels are attainable to prove against the highest model with same validation strategies.

Creating a Project:

- ✦ Begin Mentor graphics hdl designer.
- ✦ Open the terminal (Start->Accessories->Terminal)
- ✦ Set up the atmosphere by writing the following to the terminal
- ✦ $ source /share/tktprog/mentor/hds-2008.1a/hds.sh
- ✦ The HDL designer is start by writing $ hds&
- ✦ If you begin the program for the first time, press "Cancel" to exit the HDS setup assistant
- ✦ Generate a new project by giving name.
- ✦ Give the name for project and set directory.


Creating a New Project
Please specify information about the new project
Name of new project:
my_project
Optional short description:
Directory in which your project folder will be created:
D:\user\taatilao\hds_projects\HDS\tutorial1
Browse...
Name for the default working library:
tutorial1_lib
Advanced
Next >
Cancel
Help


ModuleWare Parameters
Information
N Input AND - and
Category - Logic
ANDs N inputs of the same width
Optional ports - none
Details
Name
Value
din, dout
<Automatic>
din0_type
ActiveHigh
din1_type
ActiveLow
din2_type
ActiveHigh
din3_type
ActiveLow
dout_type
ActiveHigh
ActiveLow
Options
Dynamic number of ports: 4
Preview...
Show parameters text box
Params Display...
Auto display dialog after adding part
Restore Defaults
Generate in-line
Name:
OK
Cancel
Apply
Help



Modelsim Software:

- In Modelsim, the designs are compiled into a library. You typically start a new simulation in Modelsim by making an operating library referred to as "work". "Work"is that the library name employed by the compiler because the default destination for compileddesign units.
- Compiling Your Design: when making the operating library, you compile your design units into it. You‟ll be able to simulate your design on any platform without having to recompile your design.
- Load the simulator with your design and run the simulation with the design compiled, you load the simulator together with your design by bringing the simulator on a top module.
- By assuming the design loads with success, the simulation time is ready to become zero, after that you enter a run command to start simulation.
- Debugging results, if you don‟t get the results you expect, you can use ModelSim‟s robust debugging environment to track downthe cause of the problem.

4.3.1 Introduction to Model Simulator:

Basic Simulation Flow: The below diagram gives the idea for simulating a design in ModelSim tool.

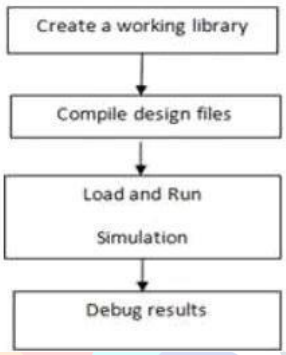


Figure Basic simulation flow

Project Design Flow: As you will saw, the flow is exact as the basic simulation flow. However, the two important dissimilarities aregiven.

- You don't have to be compelled to generate an operating library within the project flow, it's done for you automatically.
- They‟ll open on every occasion you request Modelsim unless you specifically shut them.
- The below diagram gives the basic steps for simulating a design among a Modelsim project.

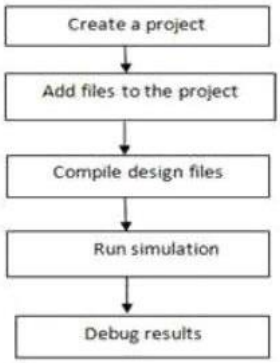


Figure Project design flow

4.3.2 Introduction to XILINX ISE:

Xilinx ISE tool can be used to designs for implement on FPGA chips.

- ISE: Integrated Software Environment
- Domain for the implementation and test of digital systems design aims to FPGAor CPLD.
- Integrated collection of tools attainable through a GUI

- Based on a logical synthesis design
- XST hold up dissimilar languages
- Verilog
- VHDL
- XST generate a net list integrated with restriction
- Supports all the steps required to complete the design
- Convert, map, place and route
- Generating a bit stream
- Bear up confirmation at different steps of the design
- In this case, to check the functionality of design test bench is written by usingfiles on the host computer.

Implementation:

- Synthesis (XST): Produce a net list file beginning from an HDL description
- Translate (NGD Build): Change all input style net lists then write the results onone integrated file, that

describes logic and constraints.

- Mapping (MAP): Map the logic on device elements. Takes a net list and teams

the logical components into CLBs and IOBs.

- Place and Route (PAR): Place FPGA cells and connects cells.

XILINX Design Process:

- Step1: Design entry – HDL (Verilog or VHDL, ABEL x CPLD), SchematicDrawings, and Bubble

Diagram.

- Step2: Synthesis – Translates .v, .vhd, .sch files into a net list file (.ngc)
- Step3: Implementation – FPGA: Translate/Map/Place & Route, CPLD: Fitter
- Step4: Configuration/Programming – Download a BIT file into the FPGA – Program JEDEC file into

CPLD – Program MCS file into Flash PROM Simulationcan occur after steps 1, 2, 3

4.4 Verilog Language:

Verilog hdl is the one in every of the two commonest Hardware Description Languages (HDL) utilized by integrated circuit (IC) designers. Another coding language is VHDL. HDL‟s permits design to be simulated earlier within the design cycle so as to rectify errors or experiment with completely different design. Architectures represented in hdl are not depends on technology, simple to implement and correct, and are normally more readable than schematics, significantly for huge circuits. Verilog are often describing styles at four levels of abstraction.

1. Algorithmic level
2. Register transfer level
3. Gate level
4. Switch level

The language additionally defines constructs that may be wort to management the input and output of simulation. Some Verilog constructs don't seem to be producing electronically. Most readers can need to produce electronically their circuits, thus non synthesizable constructs ought to be used just for test benches. These are program modules won‟t to generate input/output required to simulate the lasting of the design. The words “not synthesizable” will be used for examples and constructs as needed that do not synthesize. The types of code in most HDLs are two types;

Structural, the diagrams in structural are verbal wiring which are without memory.assign a=b | c & d; /* “|” is a OR

*/

assign d = c & (~e);

The orders of the statements are not a matter. Change in a will occur by changing in c.

Procedural which is used for circuits with memory, or as a appropriate way to write conditional logic.always @(posedge clk) // Execute

the next statement on every rising clock edge.

Count <= count+1;

For synthesis, with flip-flop storage, this type of thinking generates an excessive quantity of storage. But folks like procedural code as a result of it's typically abundant easier to put in writing, as an example, if and case statements square measure solely allowed in procedural code. As a result, the synthesizers are created which might acknowledge bound kinds of procedural code as truly combinatory. But if you digress from this vogue, beware. Your synthesis can begin to fill with remaining latches.

4.4.1 Lexical tokens:

- White space
- Comments
- Numbers
- Identifiers
- Operators
- Key words

Numbers: Number storage is defined as a number of bits, but the number values are expressed in binary, octal, decimal or hexadecimal.

Examples are 3"b110, a 4-bit number, 5"d20, (=5'b1010 0), and 16„h6ED4, (=16'd28372)

Identifiers: Identifiers are user defined words for function names, variables, block names, module names and instance names. Identifiers begin with an underscore or a letter (not with a $ or a number) and may add any number of letters, digits and underscores. The identifiers which are in verilog are case sensitive.

Operators: To perform some operations on variables operators are used. The operators are one, two and three characters.

- Arithmetic operators
- Relational operators
- Bitwise operators
- Logical operators
- Reduction operators
- Shift operators
- Concatenation operators
- Replication operators
- Conditional operators

Key words: These are the words which have special meaning in Verilog. Some examples of the key words are assign, case, while, wire, reg, and, or, nand, and module.

4.4.2 Verilog modelling:

Verilog has four types of modelling. They are

1. The switch level of modelling.

2. The gate level of modelling.

3. The Data flow modelling.

4. The Behavioral modelling.

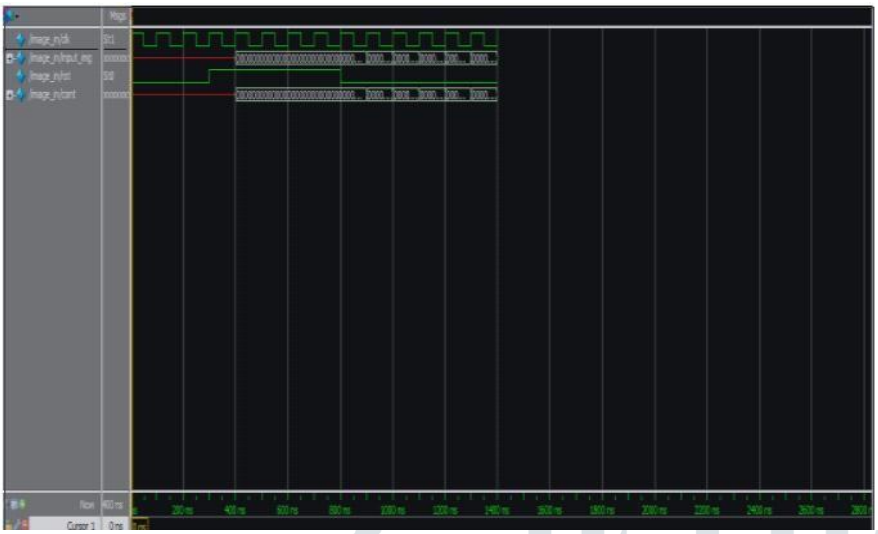
V RESULT ANALYSIS

This chapter gives the information about the output waveforms of each and every block and additionally the Xilinx results which gives the knowledge about how the area and power dissipation values of chip are dropped to a small value.

Output Waveforms:

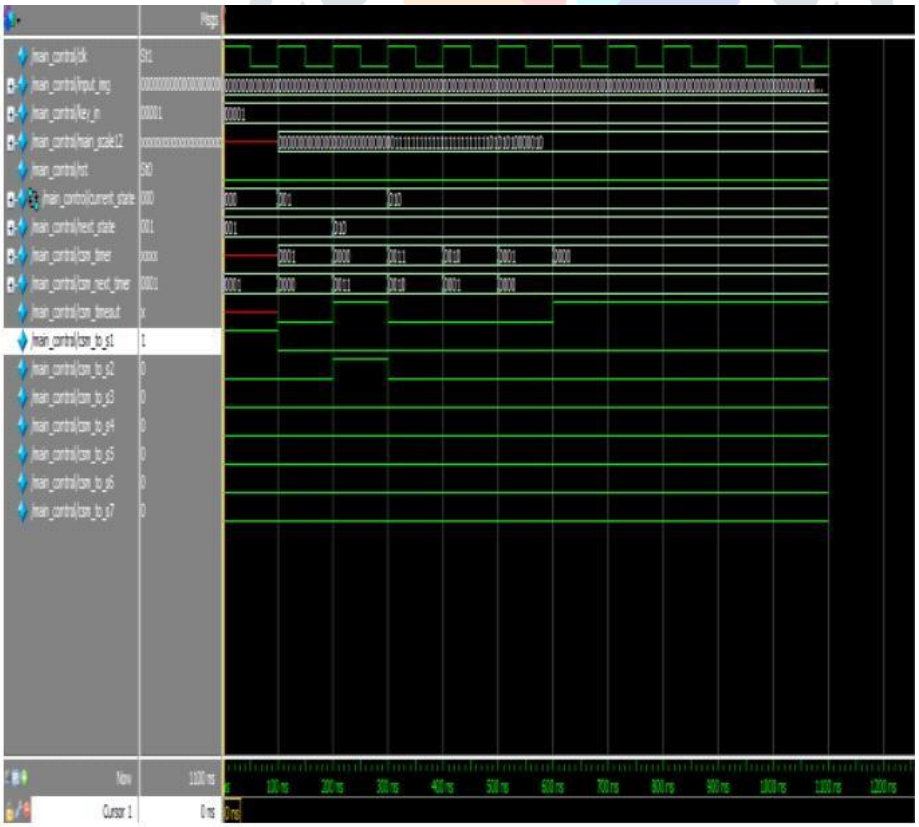
Image input block output:

This waveform output is exported from modelsim. For every clock cycle the output of image input block are changed whenever the reset value is zero. If reset value is one the output of block is not changed the value is like 000000……..



Main control block output:

In main control block output the states are changed. In this design block the key value 00001 is given then the state is altered from s0 to s1.If the key values are not given the transition of states not happen

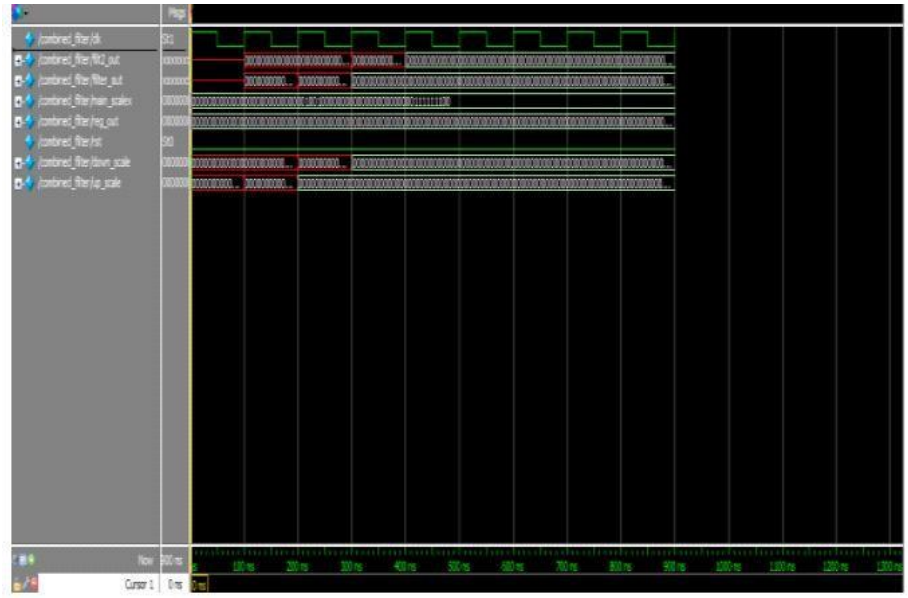


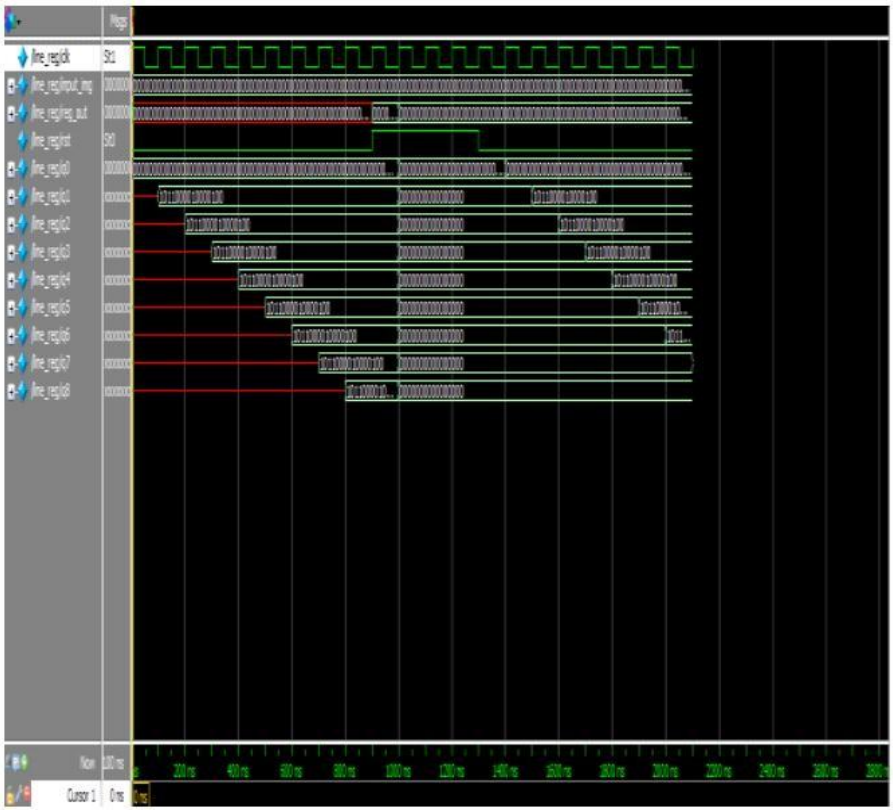
6.2.3 Line registers block output:

In line register block shifting of information bits takes place that is determined within the waveforms that takes only the reset value is zero. If reset value is one the shifting of bits not happen in waveforms

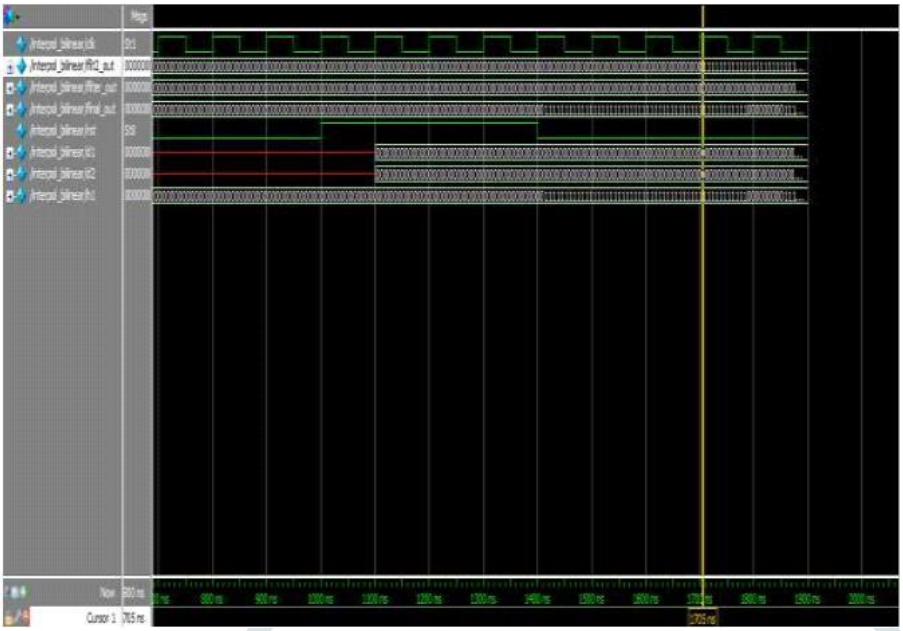
Combined filter block output:

In the combined filter block the upscaling and downscaling operations of the image takes place. The upscaled and downscaled waveforms values of image are observed here.





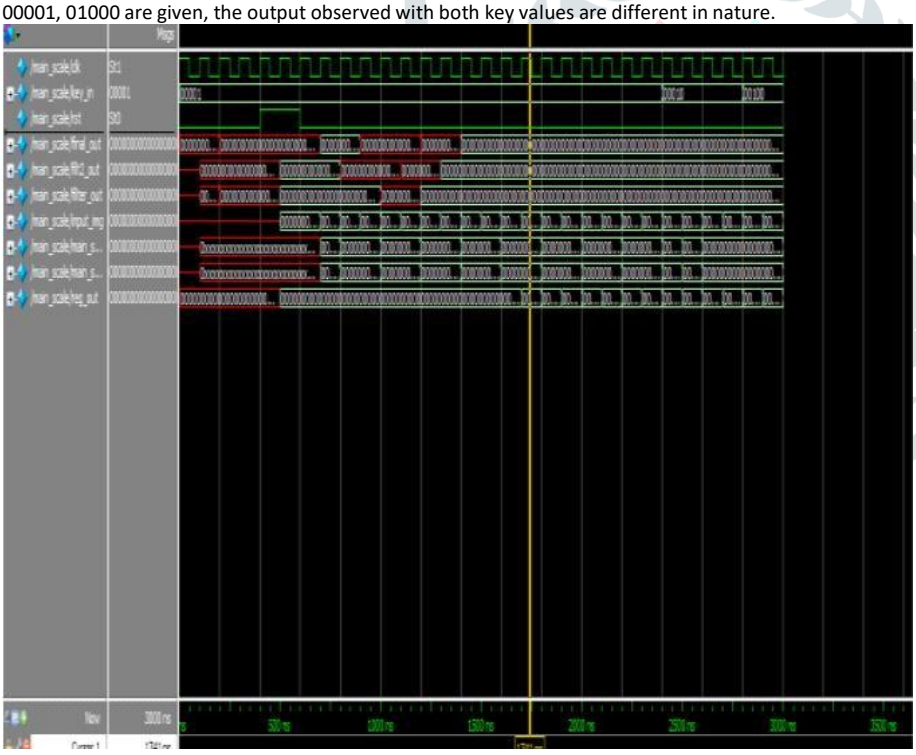
Interpol bilinear block output:



The output of Interpol bilinear block is shown in the waveform. The output image scaled values of the 2 filters are compared once comparison the larger value filter output is come into view as output

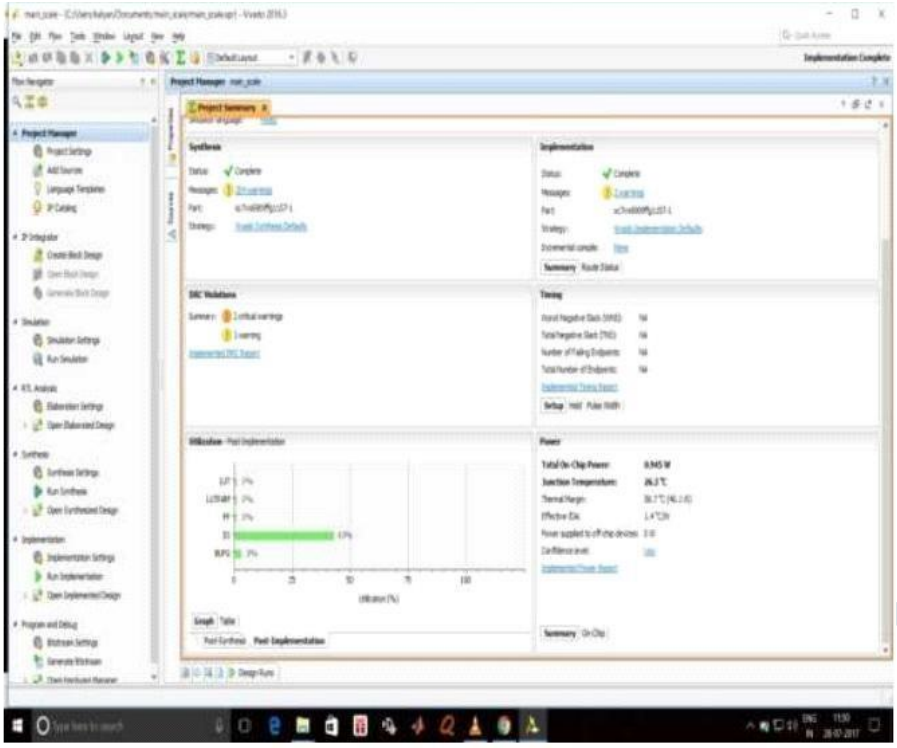
Final output:

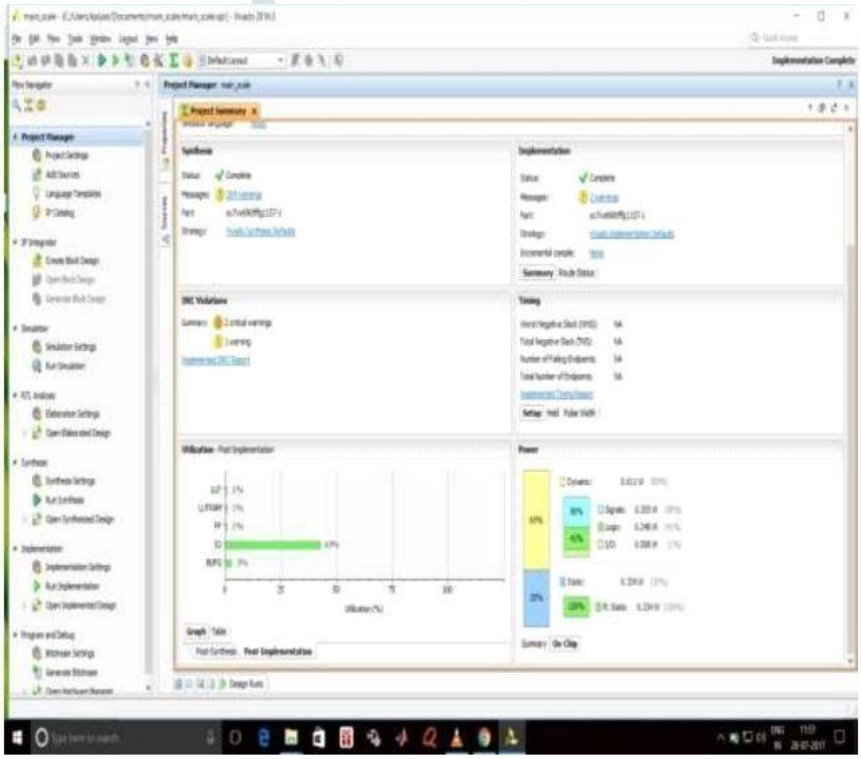
The final output waveform values are observed here whenever the key values are changed the final output values are changed. In waveform the key values 00001, 01000 are given, the output observed with both key values are different in nature.



Xilinx Results:

The Xilinx results give the information about how the power consumption is bring to a small value and area of chip is also bring to small value. The static and dynamic power dissipation values are observed. The static power ingestion is concerning 35% and dynamic power ingestion is concerning 65%. Within the dynamic power ingestion much power is dissipated in signals subsequently logics than in input output section. The on chip power is 0.945w, the junction temperature is 26.30C which is observed in Xilinx result image. The utilization of power is more in IO blocks.





CONCLUSION AND FUTURE SCOPE

This thesis confers an occasional overhead answer to implement DSP circuits which are obfuscated structurally and functionally by utilizing high level transformations techniques. It's shown that confirming the equivalence of DSP circuits by using high level transformations are tougher if some switches may be designed in such a way that they're inconvenient to trace, a configurable switch design is included within the projected design scheme to improve the protection. An entire design flow is given within the proposed

confounding methodology, the variation modes and therefore further confounded circuits might even be designed which consistently supported the high level transformations. Obfuscated and reconfigure FSM modes of which reduce the area of performance speed improved to 341.53MHZ.This work provides awareness in the foremost generalized design which will be praxis built for different image process applications too. Though, the major fundamental point performances on the image are mentioned, the thought could carried forward for filtering applications also. The key summons during this work is to settle on a correct FPGA for prototyping, since the memory buffer wants huge memory, the crucial aspect is to settle on an FPGA which has enough RAM, FIRST IN FIRST OUT resources

References

1. T. Shea, Types of Accelerators and Specific Needs, these CAS proceedings.

2. E. Angoletta, Digital Signal Processing In Beam Instrumentation: Latest Trends And TypicalApplications, DIPAC'03, Mainz, Germany, 2003.

3. J. Eyre, The Digital Signal Processor Derby, IEEE Spectrum, June 2001, pp. 62–68.

4 .Efficiently Interfacing Serial Data Converters To High-Speed DSPs, Analog Application Journal, August 2000, pp. 10–15.

5. TMS320C6000 Optimizing Compiler – User's Guide, Texas Instruments Literature NumberSPRU187L, May 2004.

6. TMS320C6000 Assembly Language Tools – User's Guide, Texas Instruments LiteratureNumber SPRU186N, April 2004.

7. TMS320C6000 Peripherals – Reference Guide, Texas Instruments Literature Number SPRU109D, February 2001.

Vinay K. Ingle, "Digital Signal Processing and Its Implementation of MATLBA", Xi'an Jiaotong University Press, 1998.

8. Meng Wenhan. Research on the application of digital signal processing technology in the communication field[J]. Electronic World, 2018, 552 (18): 124-127.

9. Zhang Ze. Application analysis of modern digital signal processing technology in optical access network [J]. Electronic Test, 2015 (09): 102-104 + 107.

10. Sun Bing. Research on the application and development of digital signal processing technology[J]. Information and Communication, 2015 (7): 177-177