A

MAJOR PROJECT STAGE -1 REPORT ON

VERIFICATION OF UART AND I2C PROTOCOL USING SYSTEM VERILOG

Submitted in partial fulfillment of the requirement for the award of degree of

BACHELOR OF TECHNOLOGY

IN

ELECTRONICS AND COMMUNICATION ENGINEERING

SUBMITTED BY

B. PAVAN KUMAR	**218R1A04D3**
B. SANDEEP REDDY	**218R1A04D4**
B. TEJASWI	**218R1A04D5**
B. PAVAN KUMAR	**218R1A04D6**

Under the Esteemed Guidance of

Ms. L. LAVANYA

Assistant Professor, ECE

DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA & NAAC)

Kandlakoya (V), Medchal (M), Telangana – 501401

2024-2025

CMR ENGINEERING COLLEGE

UGC AUTONOMOUS

(Approved by AICTE, Affiliated to JNTU Hyderabad, Accredited by NBA & NAAC)
Kandlakoya (V), Medchal (M), Telangana – 501401

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

CERTIFICATE

This is to certify that Major project work entitled **"VERIFICATION OF UART AND I2C PROTOCOL USING SYSTEM"** is being Submitted by **B.PAVAN KUMAR bearing Roll** No:**218R1A04D3, B.SANDEEP REDDY** bearing Roll No**: 218R1A04D4, B.TEJASWI** bearing Roll No: **218R1A04D5, B.PAVAN KUMAR** bearing Roll No: **218R1A04D6** in B.Tech IV-I semester, Electronics and Communication Engineering is a record bonafide work carried out by then during the academic year 2024-25.

INTERNAL GUIDE:

Ms. L. LAVANYA

Assistant Professor, ECE

HEAD OF THE DEPARTMENT

Dr. SUMAN MISHRA

EXAMINER

ACKNOWLEDGEMENTS

We sincerely thank the management of our college **CMR Engineering College** for providing required facilities during our project work.

We derive great pleasure in expressing our sincere gratitude to our Principal **Dr. A. S. Reddy** for his timely suggestions, which helped us to complete the project work successfully.

It is the very auspicious moment we would like to express our gratitude to **Dr. SUMAN MISHRA,** Head of the Department, ECE for his consistent encouragement during the progress of this project.

We take it as a privilege to thank our major project coordinator **Dr. T. SATYANARAYANA**, Professor, Department of ECE for the ideas that led to complete the project work and we also thank him for his continuous guidance, support and unfailing patience, throughout the course of this work.

We sincerely thank our project internal guide **Ms. L. LAVANYA,** Associate Professor of ECE for guidance and encouragement in carrying out this project work.

DECLARATION

We hereby declare that the Major project entitled **"VERIFICATION OF UART AND I2C PROTOCOL USING SYSTEM"** is the work done by us in campus at **CMR ENGINEERING COLLEGE,** Kandlakoya during the academic year 2024-2025 and is submitted as major project in partial fulfillment of the requirements for the award of degree of **BACHELOR OF TECHNOLOGY in ELECTRONICS AND COMMUNICATION ENGINEERING FRO JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, HYDERABAD.**

B. PAVANKUMAR (218R1A04D3)

B. SANDEEP REDDY (218R1A04D4)

B. TEJASWI (218R1A04D5)

B. PAVAN KUMAR (218R1A04D6)

CONTENTS

LISTOFFIGURES

ABSTRACT

Design Verification in VLSI is the most important step in the product development process. It aims to confirm that the system designed meets with the standards and requirements of the system. Verification is the process of checking whether the designed system performs all the required functionality specified in the design by writing the test bench or verification environment that contains group of classes and modules which generates input stimulus to the system and the output from that design is compared with the expected output. A communication system has set of roles those are called protocols. UART is a serial communication protocol that is used when only two devices are needed to communicate and it uses peer to peer topology.I2C stands for Inter Integrated Circuit used for communication between master and slave in which more than one slave devices or memory can be connected to a master device. System Verilog has been primarily used for the verification purposes in VLSI because it has the features of Hardware Description Languages such as Verilog and VHDL,C and C++ and functional coverage, assertion coverage, constrained randomization and supports OOPs concepts.

Keyword : Verification, Protocols, UART, I2C, System Verilo

CHAPTER-I

INTRODUCTION TO VLSI

1.1 VLSI TECHNOLOGY

VLSI Design presents state-of-the-art papers in VLSI design, computer-aided design, design analysis, design implementation, simulation and testing. Its scope also includes papers that address technical trends, pressing issues, and educational aspects in VLSI Design. The Journal provides a dynamic high-quality international forum for original papers and tutorials by academic, industrial, and other scholarly contributors in VLSI Design.

The development of microelectronics spans a time which is even lesser than the average life expectancy of a human, and yet it has seen as many as four generations. Early 60's saw the low density fabrication processes classified under Small Scale Integration (SSI) in which transistor count was limited to about 10. This rapidly gave way to Medium Scale Integration in the late 60's when around 100 transistors could be placed on a single chip.

It was the time when the cost of research began to decline and private firms started entering the competition in contrast to the earlier years where the main burden was borne by the military. Transistor-Transistor logic (TTL) offering higher integration densities outlasted other IC families like ECL and became the basis of the first integrated circuit revolution. It was the production of this family that gave impetus to semiconductor giants like Texas Instruments, Fairchild and National Semiconductors. Early seventies marked the growth of transistor count to about 1000 per chip called the Large Scale Integration.

By mid-eighties, the transistor count on a single chip had already exceeded 1000 and hence came the age of Very Large Scale Integration or VLSI. Though many improvements have been made and the transistor count is still rising, further names of generations like ULSI are generally avoided. It was during this time when TTL lost the battle to MOS family owing to the same

problems that had pushed vacuum tubes into negligence, power dissipation and the limit it imposed on the number of gates that could be placed on a single die.

The second age of Integrated Circuits revolution started with the introduction of the first microprocessor, the 4004 by Intel in 1972 and the 8080 in 1974. Today many companies like Texas Instruments, Infineon, Alliance Semiconductors, Cadence, Synopsys, Celox Networks, Cisco, Micron Tech, National Semiconductors, ST Microelectronics, Qualcomm, Lucent, Mentor Graphics, Analog Devices, Intel, Philips, Motorola and many other firms have been established and are dedicated to the various fields in "VLSI" like Programmable Logic Devices, Hardware Descriptive Languages, Design tools, Embedded Systems etc.

In 1980s, hold-over from outdated taxonomy for integration levels. Obviously, influenced from frequency bands, i.e., HF, VHF, and UHF. Sources disagree on what is measured (gates or transistors)

SSI – Small-Scale Integration (0-102)

MSI – Medium-Scale Integration (102 -103)

LSI – Large-Scale Integration (103 -105)

VLSI – Very Large-Scale Integration (105 - 107)

ULSI – Ultra Large-Scale Integration (>= 107)

VLSI Technology, Inc. was a company which designed and manufactured custom and semi-custom ICs. The company was based in Silicon Valley, with headquarters at 1109 McKay Drive in San Jose, California. Along with LSI Logic, VLSI Technology defined the leading edge of the application-specific integrated circuit (ASIC) business, which accelerated the push of powerful embedded systems into affordable products. The company was founded in 1979 by a trio from Fairchild Semiconductor by way of Synertek - Jack Balletto, Dan Floyd, and Gunnar Wetlesen - and by Doug Fairbairn of Xerox PARC and Lambda (later VLSI Design) magazine.

Alfred J. Stein became the CEO of the company in 1982. Subsequently VLSI built its first fab in San Jose; eventually a second fab was built in San Antonio, Texas. VLSI had its initial public offering in 1983, and was listed on the stock market as (NASDAQ: VLSI). The company was later acquired by Philips and survives to this day as part of NXP Semiconductors.

The first semiconductor chips held two transistors each. Subsequent advances added more and more transistors, and, as a consequence, more individual functions or systems were integrated over time. The first integrated circuits held only a few devices, perhaps as many as ten diodes, transistors, resistors and capacitors, making it possible to fabricate one or more logic gates on a single device. Now, known retrospectively as small-scale integration (SSI), improvements in technique led to devices with hundreds of logic gates, known as medium-scale integration (MSI). Further improvements led to large-scale integration (LSI), i.e. systems with at least a thousand logic gates. Current technology has moved far past this mark and today's microprocessors have many millions of gates and billions of individual transistors.

At one time, there was an effort to name and calibrate various levels of large-scale integration above VLSI. Terms like ultra-large-scale integration (ULSI) were used. But the huge number of gates and transistors available on common devices has rendered such fine distinctions moot. Terms suggesting greater than VLSI levels of integration are no longer in widespread use.

As of early 2008, billion-transistor processors are commercially available. This is expected to become more commonplace as semiconductor fabrication moves from the current generation of 65 nm processes to the next 45 nm generations (while experiencing new challenges such as increased variation across process corners). A notable example is NVidia’s 280 series GPU. This GPU is unique in the fact that almost all of its 1.4 billion transistors are used for logic, in contrast to the Itanium, whose large transistor count is largely due to its 24 MB L3 cache. Current designs, as opposed to the earliest devices, use extensive design automation and automated logic synthesis to lay out the transistors, enabling higher levels of complexity in the resulting logic functionality. Certain high-performance logic blocks like the SRAM (Static Random Access Memory) cell, however, are still designed by hand to ensure the highest efficiency (sometimes by bending or breaking established design rules to obtain the last bit of performance by trading stability)[citation needed]. VLSI technology is moving towards radical level miniaturization with introduction of NEMS technology. A lot of problems need to be sorted out before the transition is actually made.

1.2 WHY VLSI?

Integration improves the design, lowers the parasitics, which means higher speed and lower power consumption and physically smaller. The Integration reduces manufacturing cost - (almost) no manual assembly. The course will cover basic theory and techniques of digital VLSI design in CMOS technology. Topics include: CMOS devices and circuits, fabrication processes, static and dynamic logic structures, chip layout, simulation and testing, low power techniques, design tools and methodologies, VLSI architecture. We use full-custom techniques to design basic cells and regular structures such as data-path and memory.

There is an emphasis on modern design issues in interconnect and clocking. We will also use several case-studies to explore recent real-world VLSI designs (e.g. Pentium, Alpha, PowerPC Strong ARM, etc.) and papers from the recent research literature. On-campus students will design small test circuits using various CAD tools. Circuits will be verified and analyzed for performance with various simulators. Some final project designs will be fabricated and returned to students the following semester for testing. Very-large-scale integration (VLSI) is the process of creating integrated circuits by combining thousands of transistor-based circuits into a single chip. VLSI began in the 1970s when complex semiconductor and communication technologies were being developed. The microprocessor is a VLSI device. The term is no longer as common as it once was, as chips have increased in complexity into the hundreds of millions of transistors.

The first semiconductor chips held one transistor each. Subsequent advances added more and more transistors, and, as a consequence, more individual functions or systems were integrated over time. The first integrated circuits held only a few devices, perhaps as many as ten diodes, transistors, resistors and capacitors, making it possible to fabricate one or more logic gates on a single device. Now known retrospectively as "small-scale integration" (SSI), improvements in technique led to devices with hundreds of logic gates, known as large-scale integration (LSI), i.e. systems with at least a thousand logic gates. Current technology has moved far past this mark and today's microprocessors have many millions of gates and hundreds of millions of individual transistors.

At one time, there was an effort to name and calibrate various levels of large-scale integration above VLSI. Terms like Ultra-large-scale Integration (ULSI) were used. But the huge number of gates and transistors available on common devices has rendered such fine distinctions moot. Terms suggesting greater than VLSI levels of integration are no longer in widespread use. Even VLSI is now somewhat quaint, given the common assumption that all microprocessors are VLSI or better.

As of early 2008, billion-transistor processors are commercially available, an example of which is Intel's Montecito Itanium chip. This is expected to become more commonplace as semiconductor fabrication moves from the current generation of 65 nm processes to the next 45 nm generations (while experiencing new challenges such as increased variation across process corners). Another notable example is NVIDIA’s 280 series GPU.

This microprocessor is unique in the fact that its 1.4 Billion transistor count, capable of a teraflop of performance, is almost entirely dedicated to logic (Itanium's transistor count is largely due to the 24MB L3 cache). Current designs, as opposed to the earliest devices, use extensive design automation and automated logic synthesis to lay out the transistors, enabling higher levels of complexity in the resulting logic functionality. Certain high-performance logic blocks like the SRAM cell, however, are still designed by hand to ensure the highest efficiency (sometimes by bending or breaking established design rules to obtain the last bit of performance by trading stability).The original business plan was to be a contract wafer fabrication company, but the venture investors wanted the company to develop IC (Integrated Circuit) design tools to help fill the foundry.Its Caltech and UC Berkeley students, VLSI was an important pioneer in the electronic design automation industry. It offered a sophisticated package of tools, originally based on the 'lambda-based' design style advocated by Carver Mead and Lynn Conway.
VLSI became an early vendor of standard cell (cell-based technology) to the merchant market in the early 80s where the other ASIC-focused company, LSI Logic, was a leader in gate arrays. Prior to VLSI's cell-based offering, the technology had been primarily available only within large vertically integrated companies with semiconductor units such as AT&T and IBM.

VLSI's design tools eventually included not only design entry and simulation but eventually cell-based routing (chip compiler), a datapath compiler, SRAM and ROM compilers and a state machine compiler. The tools were an integrated design solution for IC design and not just point tools, or more general purpose system tools. A designer could edit transistor-level polygons and/or logic schematics, then run DRC and LVS, extract parasites from the layout and run Spice simulation, then back-annotate the timing or gate size changes into the logic schematic database. Characterization tools were integrated to generate Frame Maker Data Sheets for Libraries. VLSI eventually spun off the CAD and Library operation into Compass Design Automation but it never reached IPO before it was purchased by Avanti Corp.VLSI's physical design tools were critical not only to its ASIC business, but also in setting the bar for the commercial EDA industry. When VLSI and its main ASIC competitor, LSI Logic, were establishing the ASIC industry, commercially-available tools could not deliver the productivity necessary to support the physical design of hundreds of ASIC designs each year without the deployment of a substantial number of layout engineers. The companies' development of automated layout tools was a rational "make because there's nothing to buy" decision. The EDA industry finally caught up in the late 1980s when Tangent Systems released its Tan Cell and Tan Gate products. In 1989, Tangent was acquired by Cadence Design Systems (founded in 1988).

Unfortunately, for all VLSI's initial competence in design tools, they were not leaders in semiconductor manufacturing technology. VLSI had not been timely in developing a 1.0 µm manufacturing process as the rest of the industry moved to that geometry in the late 80s. VLSI entered a long-term technology partnership with Hitachi and finally released a 1.0 µm process and cell library (actually more of a 1.2 µm library with a 1.0 µm gate).As VLSI struggled to gain parity with the rest of the industry in semiconductor technology, the design flow was moving rapidly to a Verilog HDL and synthesis flow. Cadence acquired Gateway, the leader in Verilog hardware design language (HDL) and Synopsys was dominating the exploding field of design synthesis. As VLSI's tools were being eclipsed, VLSI waited too long to open the tools up to other fabrications and Compass Design Automation was never a viable competitor to industry leaders.

Meanwhile, VLSI entered the merchant high speed static RAM (SRAM) market as they needed a product to drive the semiconductor process technology development. All the large semiconductor companies built high speed SRAMs with cost structures VLSI could never match. VLSI withdrew once it was clear that the Hitachi process technology partnership was working. ARM Ltd was formed in 1990 as a semiconductor intellectual property licensor, backed by Acorn, Apple and VLSI. VLSI became a licensee of the powerful ARM processor and ARM finally funded processor tools. Initial adoption of the ARM processor was slow. Few applications could justify the overhead of an embedded 32 bit processor. In fact, despite the addition of further licensees, the ARM processor enjoyed little market success until they developed the novel 'thumb' extensions. Ericsson adopted the ARM processor in a VLSI chipset for its GSM handset designs in the early 1990s. It was the GSM boost that is the foundation of ARM the company/technology that it is today.

Only in PC chipsets, did VLSI dominate in the early 90s. This product was developed by five engineers using the 'Mega cells" in the VLSI library that led to a business unit at VLSI that almost equaled its ASIC business in revenue. VLSI eventually ceded the market to Intel because Intel was able to package-sell its processors, chipsets, and even board level products together.
VLSI also had an early partnership with PMC, a design group that had been nurtured of British Columbia Bell. When PMC wanted to divest its semiconductor intellectual property venture, VLSI's bid was beaten by a creative deal by Sierra Semiconductor. The telecom business unit

1.3 APPLICATIONS OF VLSI:

- Electronic system in cars.
- Digital electronics controls.
- Transaction processing system, ATM.
- Personal computers and Workstations.
- Medical electronic systems.

Electronic systems now perform a wide variety of tasks in daily life. Electronic systems in some cases have replaced mechanisms that operated mechanically, hydraulically, or by other means; electronics are usually smaller, more flexible, and easier to service. In other cases electronic systems have created totally new applications. Electronic systems perform a variety of tasks; some of them are visible while some are hidden.Personal entertainment systems such as portable MP3 players and DVD players perform sophisticated algorithms with remarkably little energy. Electronic systems in cars operate stereo systems and displays; they also control fuel injection systems, adjust suspensions to varying terrain, and perform the control functions required for anti-lock braking systems. Digital electronics compress and decompress video, even at high-definition data rates, on-the-fly in consumer electronics.Low-cost terminals for Web browsing still require sophisticated electronics, despite their dedicated function.

Personal computers and workstations provide word-processing, financial analysis, and games. Computers include both central processing units and special-purpose hardware for disk access, faster screen display, etc.

Medical electronic systems measure bodily functions and perform complex processing algorithms to warn about unusual conditions. The availability of these complex systems, far from overwhelming consumers, only creates demand for even more complex systems.

1.4 ASIC DESIGN FLOW:

As with any other technical activity, development of an ASIC starts with an idea and takes tangible shape through the stages of development as shown in Fig 1 and in Fig 2. The first step in this process is to expand the idea in terms of behavior of the target circuit.

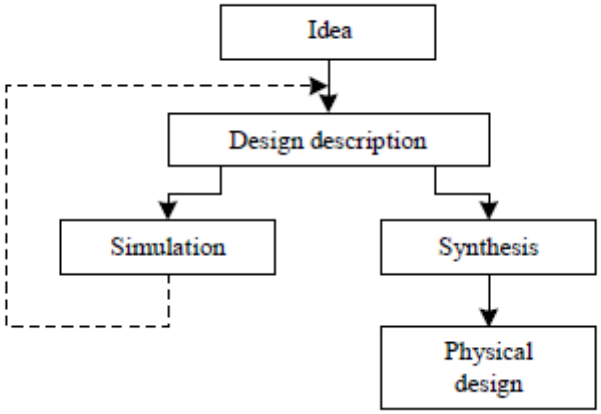


Fig 1: FLOW CHAT

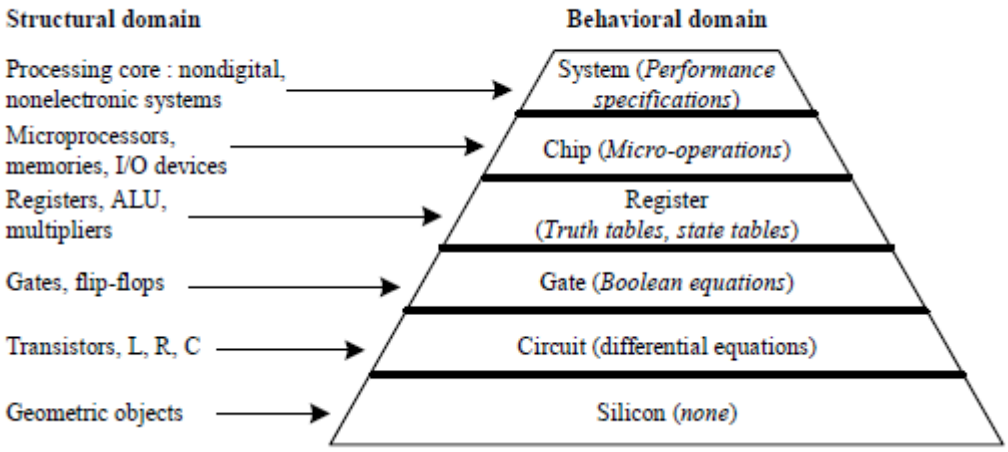


Fig 1.1: ASIC DESIGN FLOW

1.4.1 ASIC DESIGN FLOW:

The design is tested through a simulation process; it is to check, verify, and ensure that what is wanted is what is described. Simulation is carried out through dedicated tools. With every simulation run, the simulation results are studied to identify errors in the design description. The errors are corrected and another simulation run carried out. Simulation and changes to design description together form a cyclic iterative process, repeated until an error-free design is evolved. Design description is an activity independent of the target technology or manufacturer. It results in a description of the digital circuit. To translate it into a tangible circuit, one goes through the physical design process. The same constitutes a set of activities closely linked to the manufacturer and the target technology.

1.5 Verification in VLSI is Done in Two Stages

Verification is a predictive analysis process designed to ensure that a digital design will perform the specified input-output functions correctly when it is eventually manufactured [1]. It involves simulating the design to confirm it behaves as expected in all possible scenarios. Unlike testing, which is conducted on the physical product during the production phase to identify any manufacturing defects, verification focuses on the design phase, allowing engineers to detect and correct potential issues early in the development cycle. This is essential for reducing costly errors and ensuring product reliability. System Verilog, a widely adopted hardware description and verification language, offers advanced features tailored to enhance the verification process. It incorporates functionalities such as randomization, functional coverage, assertions, and Object-Oriented Programming (OOP) capabilities, all of which contribute to a more effective and flexible test bench design. These features make System Verilog a powerful tool for simulating complex digital designs and verifying that they meet their functional requirements.

1.5.1 Verification in VLSI is Done in Two Stages

Verification is a predictive analysis process designed to ensure that a digital design will perform the specified input-output functions correctly when it is eventually manufactured [1]. It involves simulating the design to confirm it behaves as expected in all possible scenarios. Unlike testing, which is conducted on the physical product during the production phase to identify any manufacturing defects, verification focuses on the design phase, allowing engineers to detect and correct potential issues early in the development cycle. This is essential for reducing costly errors and ensuring product reliability. System Verilog, a widely adopted hardware description and verification language, offers advanced features tailored to enhance the verification process. It incorporates functionalities such as randomization, functional coverage, assertions, and Object-Oriented Programming (OOP) capabilities, all of which contribute to a more effective and flexible test bench design. These features make System Verilog a powerful tool for simulating complex digital designs and verifying that they meet their functional requirements.

A test bench in System Verilog consists of a collection of classes or components, each responsible for a specific aspect of the verification process. The test bench is essential in generating and applying various stimuli to the Design Under Test (DUT), monitoring the DUT's responses, comparing expected and actual outputs, and scheduling different events such as resets and main tasks. Each class or component within the test bench is named according to its specific operation, such as stimulus generation, driving signals to the DUT, or monitoring responses [9,10]. These modular components allow engineers to structure the test bench in a way that is organized and reusable, making it easier to perform complex verification tasks across multiple design projects. References [9] and [10] highlight foundational methods for organizing and naming test bench classes, emphasizing the importance of modular and systematic approaches in modern verification environments.

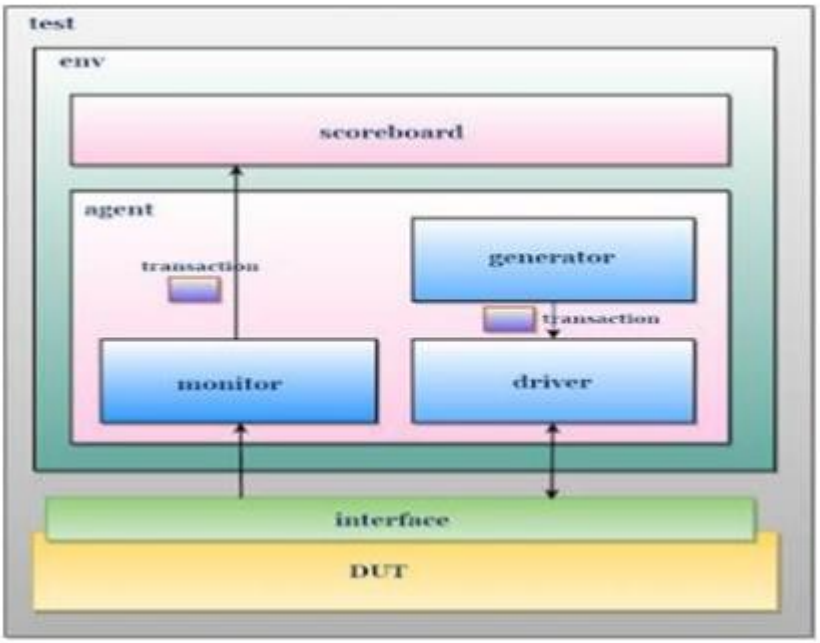


Figure 2: Block Diagram of System Verilog Test Bench

1.6 OVERVIEW OF THE PROJECT:

UART is a serial communication protocol commonly used for asynchronous data transfer between devices. It typically involves two signals: TX (Transmit) and RX (Receive). Verification of the UART protocol ensures that the transmission and reception of data work as expected, handling the timing, baud rates, and parity settings properly**.**

I2C is a multi-master, multi-slave serial communication protocol used to connect low-speed devices like sensors, EEPROMs, and microcontrollers. It typically uses two lines: SDA (Serial Data) and SCL (Serial Clock). I2C allows multiple devices to communicate using unique addresses for each device.

In conclusion, Verification of UART and I2C protocols using System Verilog involves building comprehensive testbenches that cover a wide range of conditions, including normal and error.

1.7 OBJECTIVE OF THE PROJECT:

The primary objective of verifying UART (Universal Asynchronous Receiver/Transmitter) and I2C (Inter-Integrated Circuit) protocols using System Verilog is to ensure that the hardware components and designs implementing these protocols conform to their specifications, function as intended, and handle all possible operational scenarios and error conditions effectively. Ensure that data transmitted and received via both UART and I2C is correct and matches the expected values. This includes validating that the correct data bits, start bits, stop bits, and parity bits are transferred in UART, and that acknowledgment (ACK) and data bytes are correctly transmitted and received in I2C.

Validate the baud rate settings and ensure data transmission at various baud rates, handling different data lengths, and parity configurations. Ensure error handling mechanisms, such as framing errors, overrun errors, and parity errors, are correctly implemented and flagged. Verify flow control mechanisms (if applicable), such as RTS/CTS (Request to Send / Clear to Send). Issues like data corruption, incorrect framing, overrun errors, and loss of synchronization. Simulate bus contention, address collisions, arbitration failures, and timeout errors to ensure that the protocol can recover or flag errors as needed.

Both UART and I2C protocols work correctly under various boundary conditions such as high baud rates, long data frames, extreme clock speeds, and maximum allowed device addresses. For I2C, simulate scenarios with multiple devices and high-frequency data traffic to ensure that the protocol can handle bus contention, arbitration, and message delays without errors. The implementation of UART and I2C strictly adheres to their respective standards and specifications. For I2C, ensure that devices from different manufacturers and with different addressing modes (7-bit or 10-bit) can communicate without errors, validating the interoperability of the design.

Finally, the aim is to provide the functional correctness, protocol compliance, and robustness of the design, while also identifying potential bugs, edge cases, and performance limitations. By utilizing System Verilog's advanced verification features, such as randomization, assertions.

1.8 ORGANIZATION OF THE PROJECT:

Phase 1: Structured Design:

Structured VLSI design is a modular methodology originated by Carver Mead and Lynn Conway for saving microchip area by minimizing the interconnect fabrics area. This is obtained by repetitive arrangement of rectangular macro blocks which can be interconnected using wiring by abutment. An example is partitioning the layout of an adder into a row of equal bit slices cells. In complex designs this structuring may be achieved by hierarchical nesting.

Structured VLSI design had been popular in the early 1980s, but lost its popularity later because of the advent of placement and routing tools wasting a lot of area by routing, which is tolerated because of the progress of Moore's Law. When introducing the hardware description language KARL in the mid' 1970s, Reiner Hartenstein coined the term "structured VLSI design" (originally as "structured LSI design"), echoing Edsger Dijkstra's structured programming approach by procedure nesting to avoid chaotic spaghetti-structured programs**.**

Phase 2: ASIC:

An Application-Specific Integrated Circuit (ASIC) is an integrated circuit (IC) customized for a particular use, rather than intended for general-purpose use. For example, a chip designed solely to run a cell phone is an ASIC. Intermediate between ASICs and industry standard integrated circuits, like the 7400 or the 4000 series, are application specific standard products (ASSPs). As feature sizes have shrunk and design tools improved over the years, the maximum complexity (and hence functionality) possible in an ASIC has grown from 5,000 gates to over 100 million. Modern ASICs often include entire 32-bit processors, memory blocks including ROM, RAM**,** EEPROM, Flash and other large building blocks. Such an ASIC is often termed a SoC (system-on-a-chip). Designers of digital ASICs use a hardware description language (HDL), such as Verilog or VHDL, to describe the functionality of ASICs.

Field-programmable gate arrays (FPGA) are the modern-day technology for building a breadboard or prototype from standard parts; programmable logic blocks and programmable interconnects allow the same FPGA to be used in many different applications. For smaller

designs and/or lower production volumes, FPGAs may be more cost effective than an ASIC design even in production.

- An application-specific integrated circuit (ASIC) is an integrated circuit (IC) customized for a particular use, rather than intended for general-purpose use.
- A Structured ASIC falls between an FPGA and a Standard Cell-based ASIC.
- Structured ASIC's are used mainly for mid-volume level designs.
- The design task for structured ASIC's is to map the circuit into a fixed arrangement of known cells.

CHAPTER 2
LITERATURE SURVEY

2.1 Design and Verification of UART using System Verilog

The main objective of this paper is to design and verify a full duplex UART module using System Verilog (SV). It is a serial communication protocol which provides communication between the systems without using clock signal. It converts parallel data into serial format and transmits the same. Once the data in serial format is received it is converted into parallel format. Designing of UART includes designing of baud rate generator, receiver, transmitter, interrupt and FIFO modules. Verification involves verifying the design by creating verification environment which allows to reuse the test bench and reduces the code complexity. Randomization is used to check the corner conditions which are hard to reach. 100% assertion and 100% functional coverage is achieved. UART operation is simulated using Questasim software.

2.2 I2C Protocol and its Clock Stretching Verification using System Verilog and UVM

Present day's technology has reached a goal where an entire system can be implemented on a single chip which is nothing but called system on chip (SOC). It involves microcontrollers and various peripheral devices with each peripheral device having its own intellectual property (IP) named as IP cores. Serial Communication is established between these IP cores using various protocols like RS232, RS422 and UART etc. These protocols perform point to point communication which requires huge wiring connections, multiplexing of all the bus connections to deliver the information to the IP Cores. To overcome this I2C protocol is developed by Philips, which is a two line communication. Here only two pins i.e., SCL, SDA establish connection between various devices considering one as master and other as slave, as in [1]. These two pins communicate using particular commands like start, address, read/write, acknowledgement and stop commands. Both 7-bit and 10-bit addressing formats can be used, 10-bit addressing supports more addressing lines i.e., 1024 compared to 127 addressing lines in 7-bit

2.3 Universal Verification Methodology Based Verification of UART Protocol

Verification today acts as a constriction of any complex VLSI design. Bringing out improved verification efficiency is a must. Most of the computers and microcontrollers contain a number of serial data ports. These data ports are used to connect with devices such as keyboards and printers which are basically serial input and output devices. Transmission and reception of serial data from an isolated location can be done with the help of a modem connected to the serial port. UART- Universal Asynchronous Receiver and transmitter is a hardware device which facilitates serial transmission and reception of data. In this work presented here, the UART has been designed with the use of the industry standard Verilog HDL code and the verification of the protocol has been done using system Verilog code in UVM environment. The UVM based verification methodology can significantly reduce the time needed for verification.

2.4 Design and Implementation of I2C Communication Protocol on FPGA for EEPROM

The I2C protocol was given by Philips Semiconductors in order to allow faster devices to communicate with slower devices and also allow devices to communicate with each other over a serial data bus without data loss. We here present a model of I2C bus controller, the I2C controller is designed using Verilog HDL in Xilinx 12.2. The EEPROM, ADC and RTC will require an interface for communication between them. So I2C bus is used as an interface between them. So it is used to minimize system level interconnect. More over transmitting information over the I2C bus will improve system performance, since the transmission of digital data is much less susceptible to interference from environmental noise sources. To implement the I2C protocol on an FPGA for interfacing with an EEPROM, the design includes a finite state machine (FSM) to manage start/stop conditions, addressing, data transfer, and acknowledgment handling. A clock divider generates the required SCL frequency from the FPGA's internal clock. The EEPROM is accessed by first transmitting its device address, followed by a memory address, and then data for read or write operations.The design can be developed using Verilog or VHDL and simulated using tools like ModelSim to validate functionality. Once verified, it is synthesized and programmed FPGA.

2.5 Review on Performance Analysis of UART

A Universal Asynchronous Receiver/Transmitter, abbreviated UART, is a type of "asynchronous receiver/transmitter", a piece of computer hardware that translates data between parallel and serial forms. UARTs are commonly used in conjunction with communication standards such as EIA RS-232, RS-422 or RS-485. A UART is usually an individual (or part of an) integrated circuit used for serial communications over a computer or peripheral device serial port. Modern ICs now come with a UART that can also communicate synchronously. he Universal Asynchronous Receiver/Transmitter (UART) is a widely used hardware communication protocol for serial data transmission. It is often employed in embedded systems, microcontrollers, and communication modules due to its simplicity and flexibility. The performance of UART is typically evaluated in terms of baud rate, data integrity, latency, and error handling.

UART communication transmits data in a structured frame that includes a start bit, data bits (typically 5–8 bits), an optional parity bit for error checking, and one or two stop bits. The baud rate determines the data transmission speed, measured in bits per second (bps), and must be synchronized between the sender and receiver for accurate communication. Common baud rates include 9600, 19200, and 115200 bps.

2.6 EXISTING ARCHITECTURE:

The document describes the architectures for UART and I2C protocols.

UART Architecture: This is represented with a transmitting UART and receiving UART. The block diagram typically includes:

Transmission line (Tx) and reception line (Rx).

Start and stop bits with data frames.

Parallel-to-serial and serial-to-parallel data conversion components.

Generally, there are two types of UARTS one is transmitting UART and other is entering UART and the commerce between these two can be done directly [4, 5, 6]. For this, simply two lines are needed to communicate between two UARTs. The inflow of data will be from both the transmitting(Tx) & receiving(Rx) legs of the UARTs. In this protocol the Communication from Tx UART to Rx UART can be done without timepiece. In UART the data transmission can be done like a microcontroller, memory, CPU[7]. The entered UART data packet contains three bits like launch, stop and equality in addition to entered data. It reads the data packet bit by bit and converts the entered data into the resemblant form to exclude the three bits of the data packet. The data packet entered by the UART transfers in resemblant to the entering end by data machine.

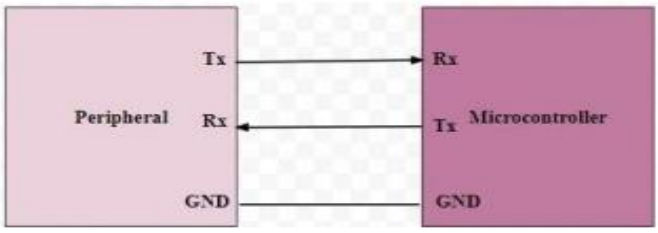


Figure 3: UART communication

I2C Architecture: This includes:

Master and slave configuration with data lines (SCL for the clock, SDL for data).

An address frame followed by data frames.

Acknowledge (ACK) or negative acknowledge (NACK) signals.

I2C is a multi-master and multi-slave periodical communication protocol means that we can attach multiple IC at a time with the same machine[2]. In the I2C protocol, the master has control on motorcars and in the case of multi-master, only one master will control I2C machine. The data transfer and synchronization between the master and slave is done by timepiece signal. In communication master and slave partake the same timepiece, for coetaneous periodical communication. The timepiece machine is handled by the master but in some conditions the slave is also suitable to control the timepiece[3].

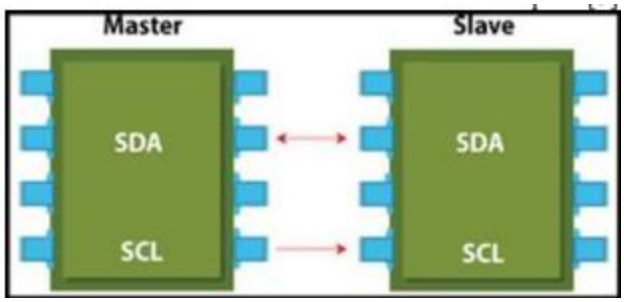


Figure 4: I2C communication

2.1 PROPOSED ARCHITECTURE:

System Verilog Verification Architecture:

Test Bench Environment:

Components:

Stimulus Generator: Generates test vectors or input data.

Driver: Sends input data to the DUT (Device Under Test).

Monitor: Observes output data from the DUT and logs it.

Scoreboard: Compares observed outputs against expected results for verification.

DUT:

The implementation of the UART or I2C module being tested.

Illustration:

A comprehensive diagram showing the test bench environment with each component (Generator, Driver, Monitor, DUT, Scoreboard) interconnected to mimic verification flow.

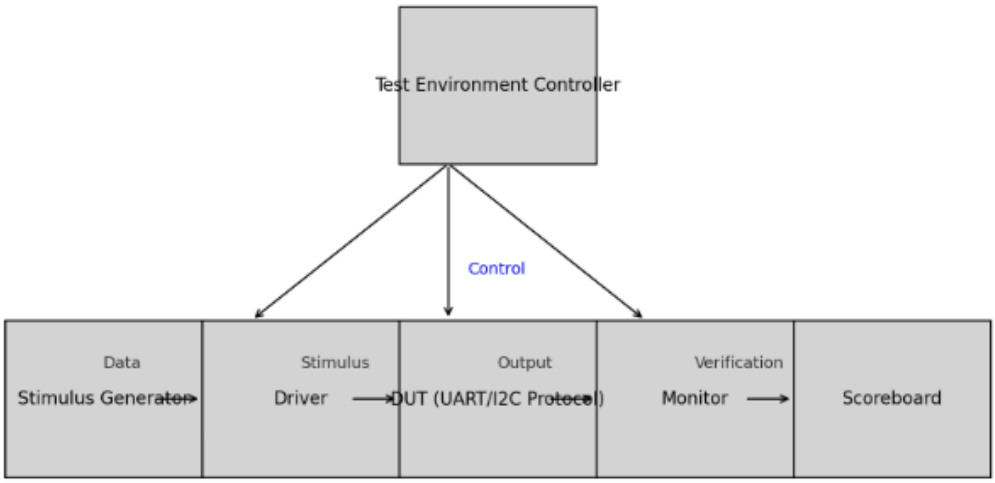


Fig 5: Proposed System

CHAPTER 3

HARDWARE AND SOFTWARE REQUIREMENTS

3.1 SOFTWARE REQUIREMENTS:

- **Xilinx Vivado:** Popular for FPGA development with Verilog and VHDL, especially for Xilinx boards.

- **Operating System :** Windows or Linux:

3.2 XILINX

Xilinx, Inc. is the world's largest provider of programmable common sense devices, the inventor of the field programmable gate array (FPGA) and the primary semiconductor organization with a fabless manufacturing version. Xilinx designs, develops and markets programmable logic merchandise including incorporated circuits (ICs), software program design equipment and predefined gadget functions added as intellectual property (IP) cores, design offerings, patron education, area engineering and technical assist. Xilinx sells each FPGAs and CPLDs programmable common sense devices for electronic equipment producers in cease markets along with communications, commercial, customer, automobile and statistics processing.

Xilinx's FPGAs have even been used for the ALICE (A huge Ion Collider test) on the CERN ecu laboratory at the French-Swiss border to map and disentangle the trajectories of heaps of subatomic debris. The Vertex-II seasoned, Virtex-6, Virtex-five, and Virtex-6 FPGA families are mainly focused on gadget-on-chip (SOC) designers due to the fact they consist of up to two embedded IBM PowerPC cores. The ISE layout Suite is the critical digital design automation (EDA) product own family sold by using Xilinx. The ISE design Suite features include design access and synthesis assisting Verilog or VHDL, place-and-route (PAR), completed verification and debug using Chip Scope pro equipment, and advent of the bit documents which can be used to configure the chip. XST-Xilinx Synthesis era performs device particular synthesis for Cool Runner XPLA3/-II and XC9600/XL/XV households and generates an NGC report ready for the CPLD more fit.

Xilinx is a prominent semiconductor company recognized for its cutting-edge programmable logic devices, including Field Programmable Gate Arrays (FPGAs) and Complex Programmable Logic Devices (CPLDs). Established as a leader in the FPGA market, Xilinx offers highly customizable hardware solutions that can be reprogrammed to meet specific design requirements even after manufacturing, making them ideal for a variety of applications.

Xilinx's flagship technology includes FPGAs, which provide designers with the flexibility to program and configure logic to implement complex digital systems. Some of the key FPGA families from Xilinx include the Spartan, Virtex, and Kintex series, each designed for different performance, power, and cost requirements. Additionally, Zynq SoCs (System on Chips) integrate FPGA fabric with ARM-based processors, allowing for more efficient hardware/software integration, commonly used in embedded systems.

Xilinx has also pioneered the Adaptive Compute Acceleration Platform (ACAP), a revolutionary technology combining programmable logic, processor cores, and AI engines on a single chip. This enables high-performance applications, particularly in areas like artificial intelligence, machine learning, and real-time data processing. Furthermore, the Versal Adaptive SoCs mark the next generation of Xilinx devices, blending programmable logic with scalable processing, AI, and networking capabilities, opening new possibilities in data centers, automotive, and industrial sectors.

Overall, Xilinx's programmable solutions are integral to many industries, including telecommunications, aerospace, automotive, and consumer electronics, offering flexibility, performance, and scalability for modern technological advancements.

3.3 XILINX ISE 13.2i:

Xilinx is the maximum important tool and in this device we are able to carry out both simulation and synthesis.

3.3.1 Simulation:

In this process, we are going to verify our required output to get the simulation technique first of all we need to enforce a top module (combination of all modules) after which in the simulation conduct we can simulate the result

3.3.2 Synthesis:

Synthesis process defines converting Verilog code into gate level which creates a net list.

3.3.3 Procedure:

- Click project navigator
- Create new project
- Selection of FPGA

Create new source

- Select source type (Verilog module)
- Coding
- Declaration of inputs and output
- Sources for implementation

Synthesize – XST

- Check syntax
- View design summary
- View RTL schematic
- View technology schematic

3.4 Verilog HDL:

Verilog is one amongst the chief regular Hardware Description Languages (HDL) utilized by PC circuit (IC) architects. HDL's licenses style to be recreated before inside the outline cycle in order to right mistakes or try different things with totally diverse models. Styles spoke to in lipoprotein zone unit innovation free, easy to style and redress, and range unit now and then a considerable measure of decipherable than schematics, fundamentally for enormous circuits.

3.4.1 MODELING TECHNIQUES:

3.4.1.1 Data stream:

In this form we can depict the parts without a moment's delay by the connection among them.
Module is a catchphrase which signifies the relationship between particular components with The module call is and entryway that is having data sources and yields are pronounced as in the bracket.The dole out statement is a watchword which indicates plays out the operation particular. At that point it will spare the expense in the left hand aspect operand. endmodule proclamation means that completing touch of module

3.4.1.2 Behavioral:

That is the demonstrating method that is utilized to characterize the element without knowing it. We can adaptation the behavior. We can outline the issue by method for its conduct best. The dependably watchword recommends a free running technique. This proposes zero running test system. When an always obstruct achieved its give, it's far rescheduled (yet again). Parameters inside the Parenthesis are called affectability posting. The affectability list which recommends while information sources are assessed then constantly square can be accomplished.
The if else proclamation is comparable like as in C. while the separate affirmation is right comparing outcomes may be expert.

3.4.1.3 Structural demonstrating:

That is utilized to format an intricate modules the utilization of straightforward sub module of it. The sub modules or the added substances which can be utilized routinely inside the bigger applications. These techniques will make complex applications yet basic design.

3.4.2 MODULES :

In Verilog, circuit added substances are planned inside a module. Modules can fuse both basic and behavioral explanations. Auxiliary proclamations constitute circuit segments like rationale doors, counters, and chip. Behavioral degree proclamations are customizing explanations that have no immediate mapping to circuit added substances like circles, if-then articulations, and jolt vectors which may be utilized to practice a circuit. Underneath code demonstrates an occurrence of a circuit and an investigate seat module. A module begins off developed with the watchword module joined by utilizing a non-mandatory module name and a non-necessary port rundown. The essential thing phrase end module closes a module.

3.4.3 STRUCTURAL DESIGN WITH GATE AND DELAY OPERATOR

Verilog characterizes some essential practical insight entryways as a part of the dialect. Module some rationale part instantiates two door primitives: the not entryway and the AND entryway. The yield of the door is the primary parameter, and the information sources are whatever remains of the parameters. These primitives are versatile with the goal that you can get more than one information entryways basically by means of including contributions to the parameter posting.

3.4.4 STRUCTURAL DESIGN WITH ASSIGNMENT STATEMENTS

On the off chance that you have many irregular great judgments, the door primitives of the previous segment are dreary to utilize in light of the fact that all the inner wires should be announced and set up viably. Every once in a while it's miles less muddled to simply depict a circuit the use of an unmarried Boolean condition. In Verilog, Boolean conditions that have practically identical planning houses as the door primitives are portrayed the use of a constant mission statement.

3.4.5 STRUCTURAL DESIGN WITH USING MODULES

Verilog helps progressive design by utilizing allowing modules to instantiate different modules. As a matter of course the planning inside a module is controlled through the module itself. However, modules can be characterized to have parameterized delays like the #(6,five) put off administrator utilized with entryway primitives. In the module definition, utilize the parameter watchword to make puts off variables. Parameters additionally can be utilized to change other scalar qualities in the module. At the point when the module is instantiated then you could supersede the put off qualities utilizing the #(parameter) documentation.

3.4.6 BEHAVIORAL DESIGN WITH INITIAL AND ALWAYS BLOCKS

Behavioral code is utilized to portray circuits at a more dynamic level then the basic level explanations we have concentrated on. All Behavioral code happens inside either an underlying square or in a generally piece. A module can contain various preparatory and continually pieces. These behavioral pieces consolidate explanations that oversee reenactment time, realities take the path of least resistance articulations (like assuming then and case proclamations), and closing off and non-blocking proclamations.

• An beginning piece executes when all through a reenactment. Preparatory pieces are for the most part used to instate variables and to clarify jolt waveforms which exercise which weight the reproduction.

• An ordinarily piece continually rehashes its execution for the span of a recreation. Consistently squares generally join behavioral code that models the genuine circuit operation.

All through a recreation each dependably and each preparatory square start to execute at time zero. Each piece executes at the same time with each auxiliary articulation and all the distinctive behavioral squares. The accompanying example shows a behavioral SRAM form. The underlying square units the memory cells to 0 at startup. The always square executes at whatever point there's a substitute on the compose oversee line, the chip pick line, or the location transport. As a workout, propagation and glue this code into a Verilog document and compose a test seat to practicing the model

3.4.7 STRUCTURAL DATA TYPES: WIRE AND REG

Verilog bolsters basic data sorts alluded to as nets which model equipment associations between circuit added substances. The two most regular auxiliary data sorts are wire and reg. The string nets act like genuine wires in circuits. The regtype keep their qualities till some other expense is set on them, much the same as a check in equipment thing. The assertions for twine and enroll pointers are inside a module however open air any underlying or typically square. The preparatory condition of a register is x obscure, and the preparatory country of a twine is z. Ports: Modules speak with each other through ports, the cautions ordered in the parameter list at the highest point of the module. Ports can be of sort in, out, and in out. Right here are three oversimplified controls for coordinating the basic records kind to the sort of port:

3.4.8 BEHAVIORAL DATA TYPES: INTEGER, REAL, AND TIME

The sorts in whole number and genuine are helpful records sorts to apply for checking in behavioral code pieces. Those data sorts act like their counter parts in other programming dialects. on the off chance that you at some point or another arrangement to blend your behavioral code you then could likely need to abstain from utilizing these records sorts because of the reality they habitually combine huge circuits. The information kind time can safeguard a unique test system esteem known as reenactment time which is extricated from the gadget trademark $time. The time insights might be utilized to help you investigate your recreations. In hardware description languages (HDLs) like Verilog and VHDL, behavioral data types are used to model and simulate various attributes of digital systems at an abstract level. These data types include integer, real, and time, each serving specific purposes in modeling and simulation

3.4.9 NUMBER SYNTAX

Numbers in Verilog are inside the accompanying arrangement the scale is always exact as a decimal reach. On the off chance that no is exact then the default length is no less than 32bits and can be expansive relying upon the gadget. legitimate base configurations are 'b , 'B , 'h , 'H 'd , 'D , 'o , 'O for twofold, hexadecimal, decimal, and octal. Numbers comprise of series of digits (0-9, A-F, a-f, x, X, z, Z). The X's mean obscure, and the Z's recommend unreasonable impedance If no base design is itemized the wide assortment is accepted to be a decimal amount.

3.4.10 BEHAVIORALDESIGN 10 WITH BLOCKING ANDNONBLOCKING STATEMENTS

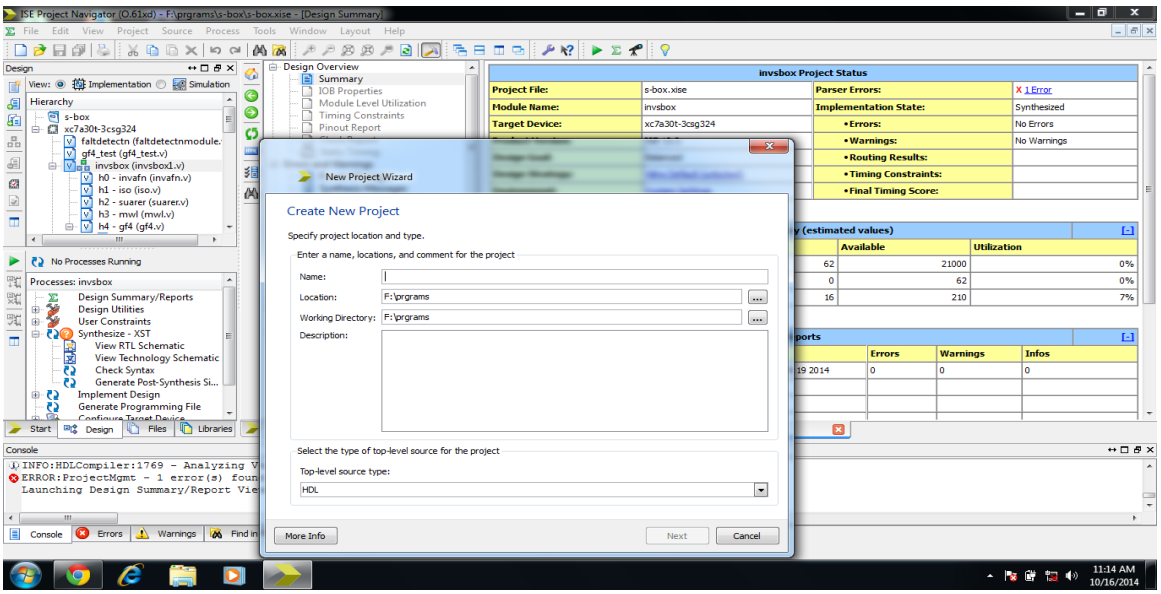
There are 2 types of undertaking articulations: hindering the utilization of the = administrator and non-barricading the use of the <= administrator. Closing off assignments act like successive code proclamations and execute while they are known as. Non-blocking time table occasions to happen at a while inside what's to come. This will be troublesome because of the reality strains that show up after a non-closing off attestation execute at the equivalent time as the non-blocking statement. Here are a couple of illustrations:

3.4.11 ARRAYS, VECTORS, AND MEMORIES

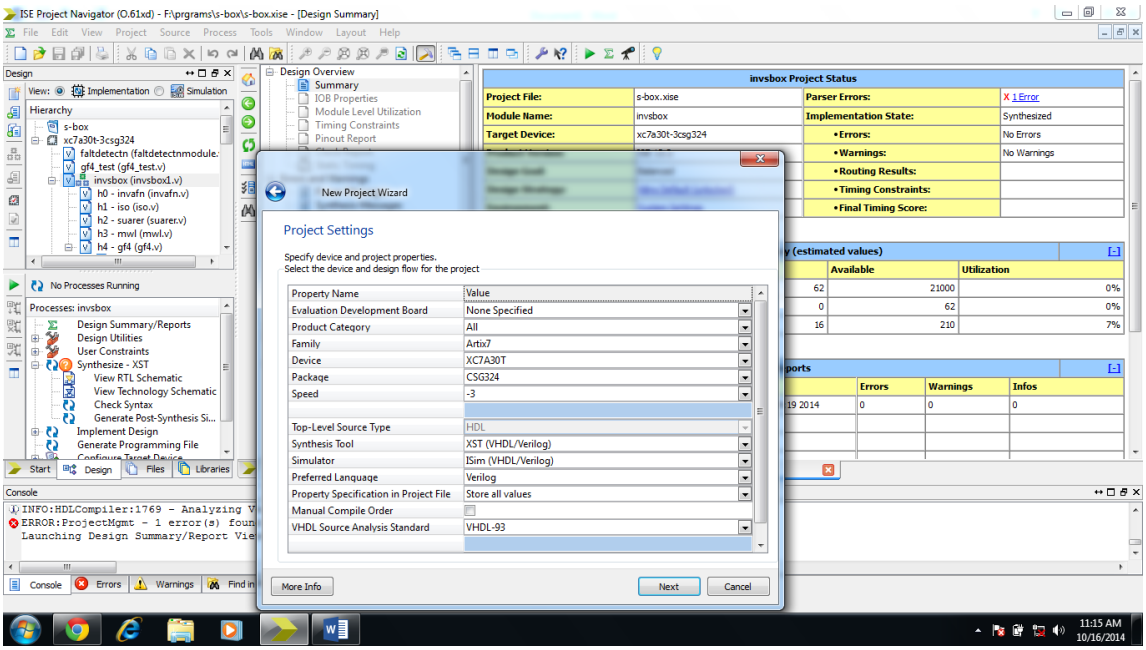
Verilog underpins three comparable measurements frameworks alluded to as Arrays, Vectors, and recollections. Clusters are utilized to save various things of the same sort. Vectors are utilized to symbolize multi-bit transports. What's more, memories are varieties of vectors which can be gotten to much like equipment recollections. Look at the accompanying case to choose an approach to reference and utilize the unprecedented actualities structures.

3.5 PROCEDURE FOR SYNTHESIS:

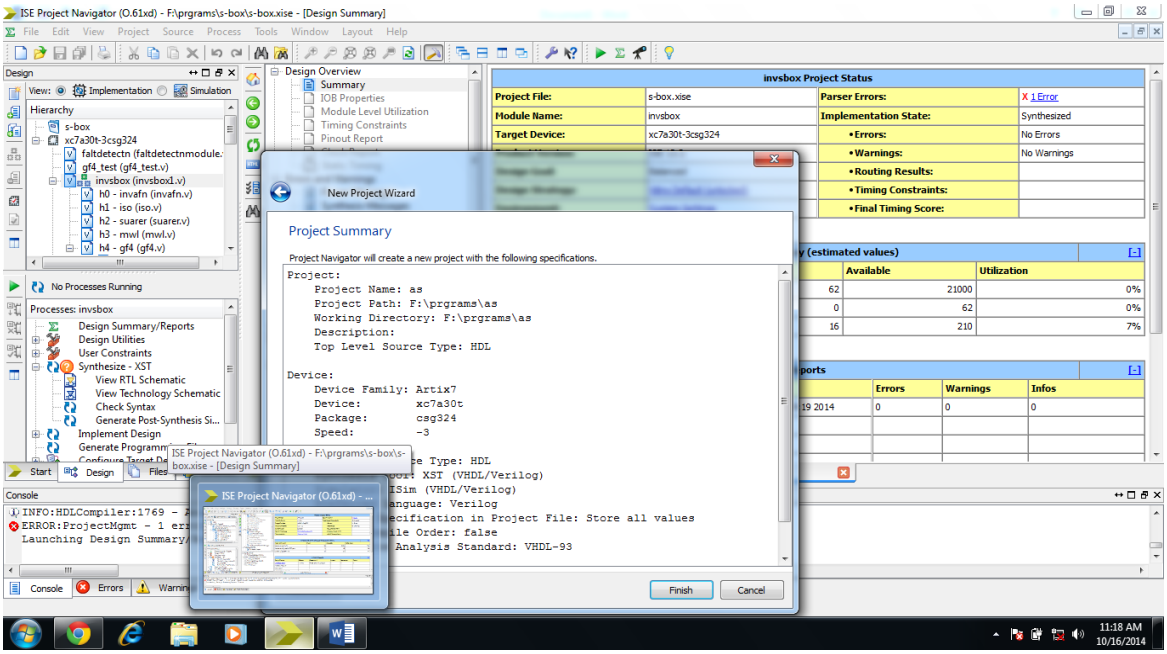
1.To create new project in xilinx we should open the filemenu,click on new project then it will open the dialogbox as below in that type the filename click on next



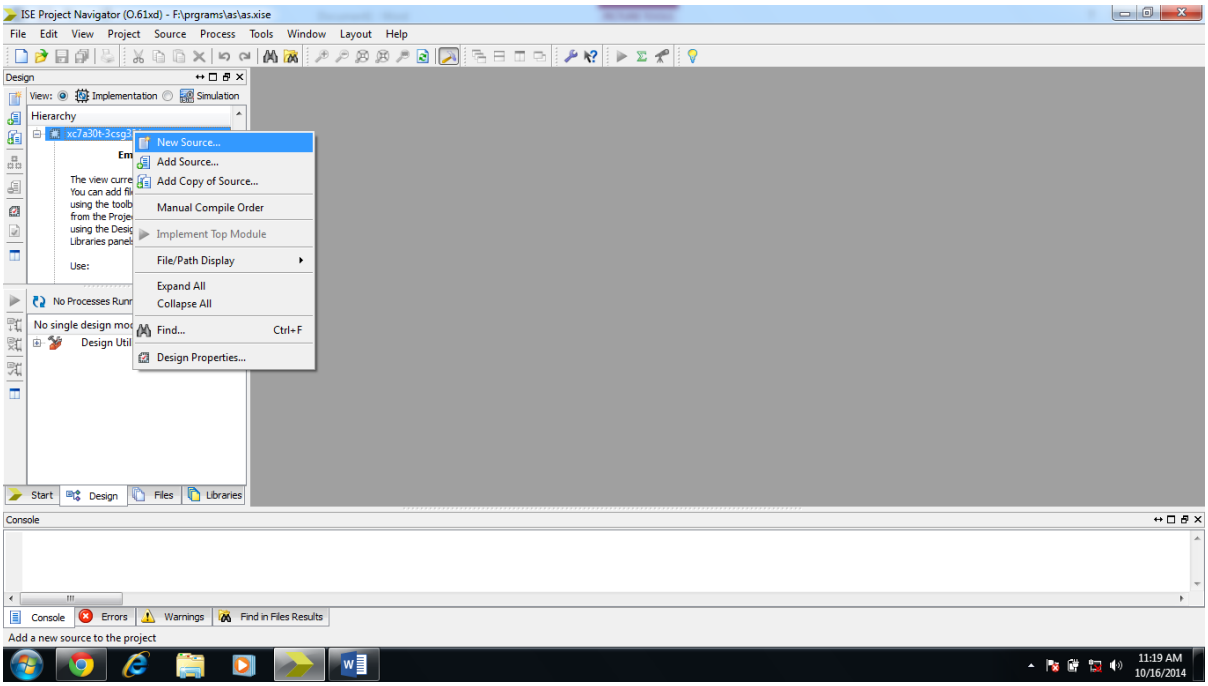
2.Then it is plays one more dialog box which will give us the specifications of the project, click on next



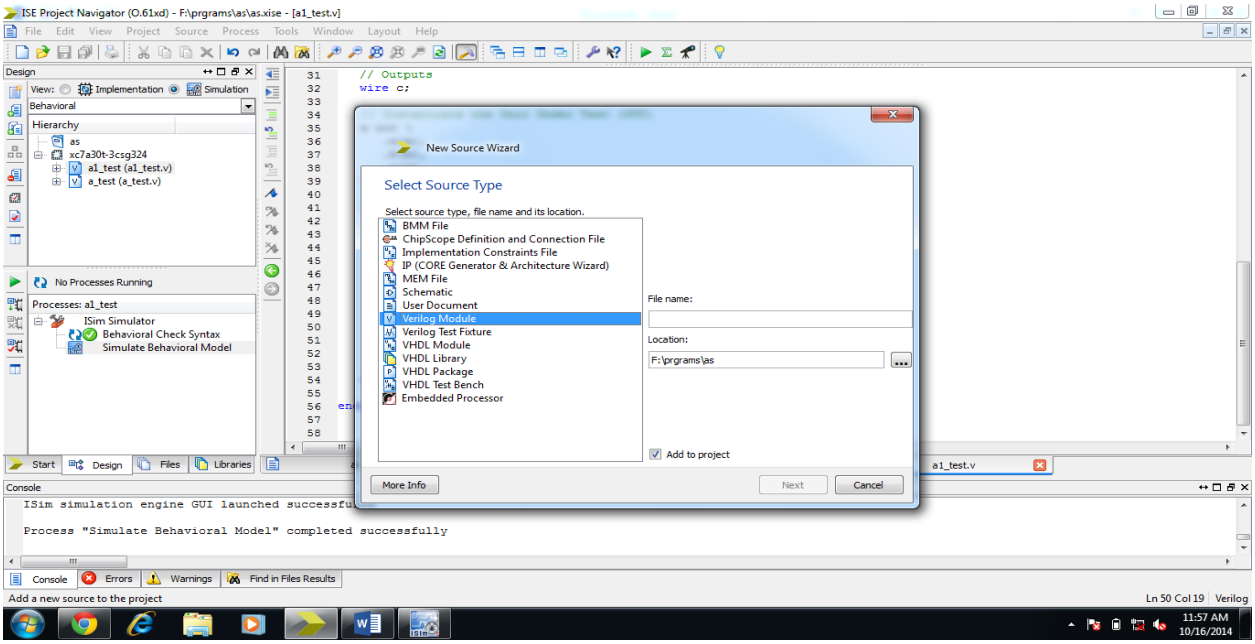
3.Then it again displays a dialogue box as shown below with the created project description and click finish to compelte the process of creating new project



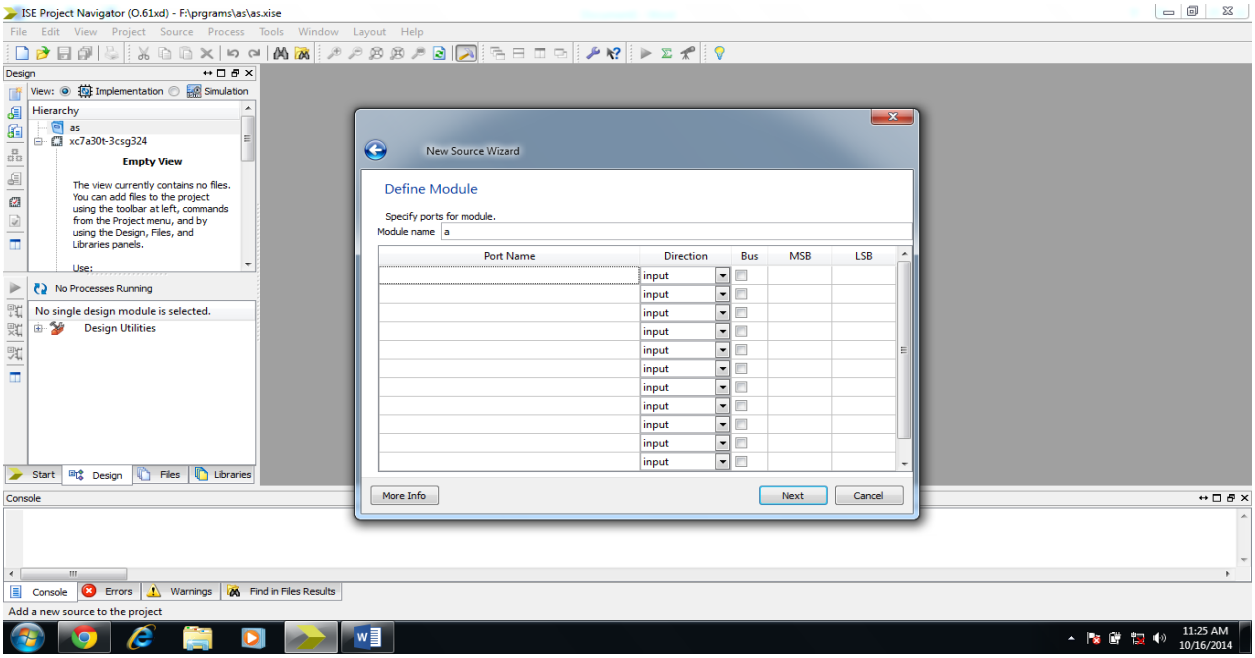
4.Now project with specifyed name is created then create the verilog files in the project. To create filesr, right click on the project that will show options like as shown below



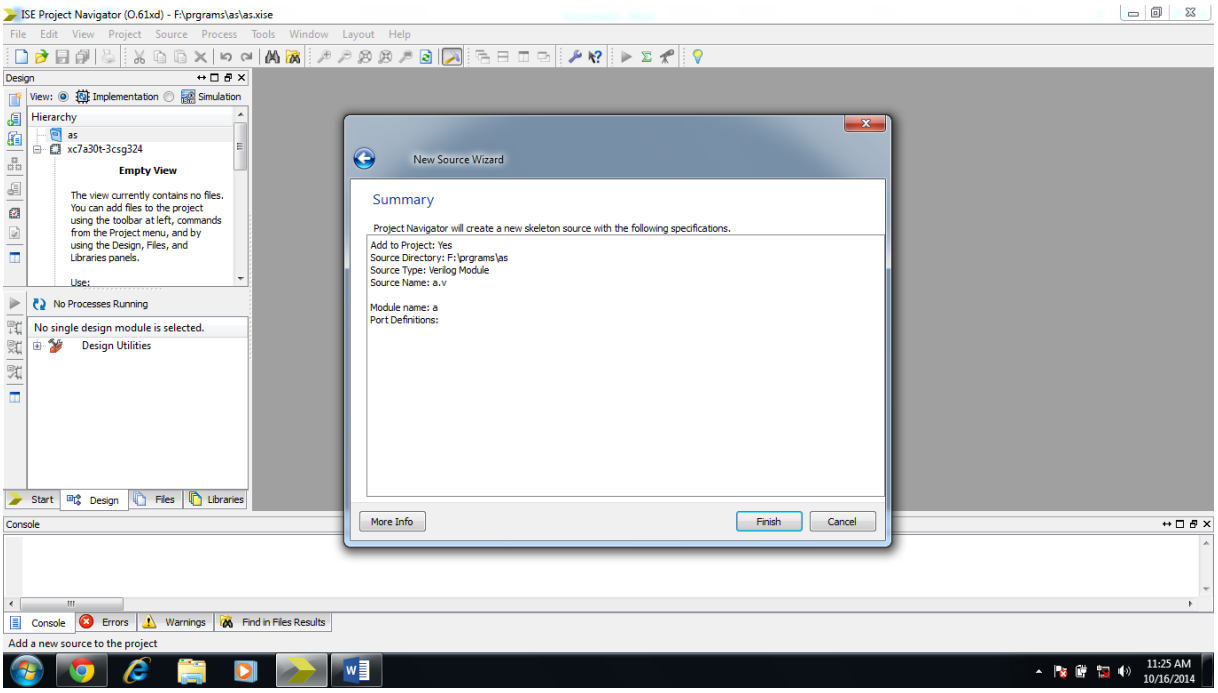
5.From the given options select new source then it diaplays dialogbox which is containing of list of fileformat now we want to create verlogfile so select veilog module.



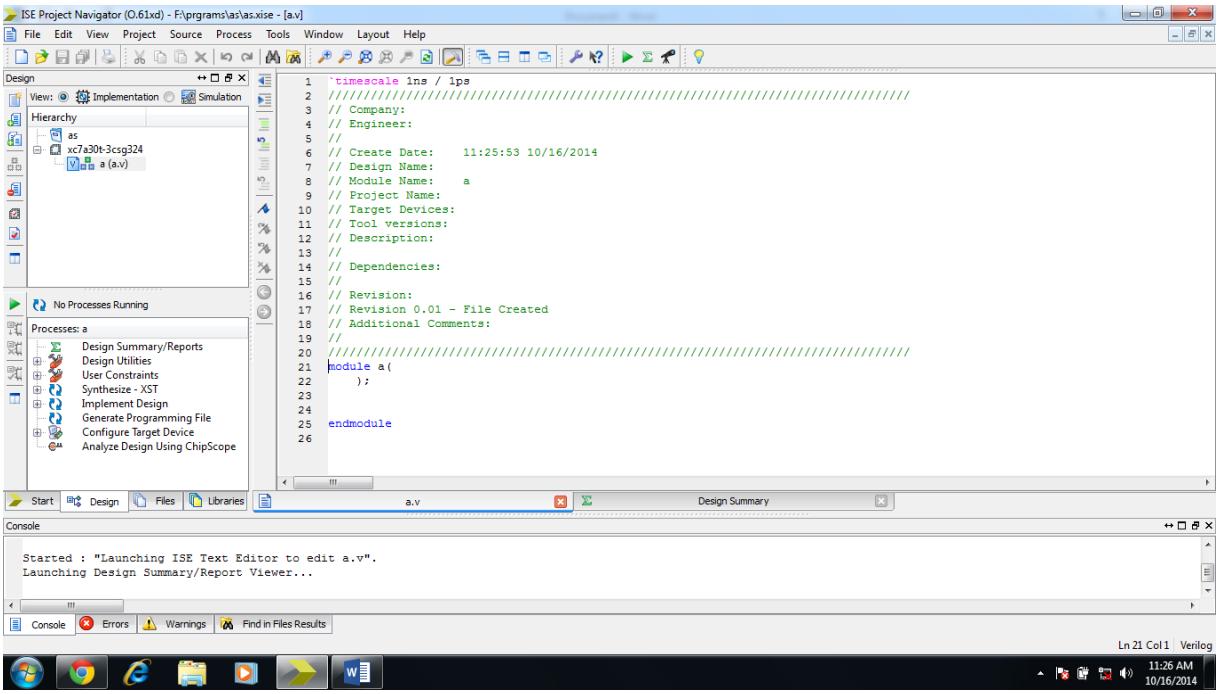
6.Then it will ask us to select inputs, outputs and in outs. We can specify our inputs and outputs here else we may also specify as part of programme depend upon the user requirement, click on next



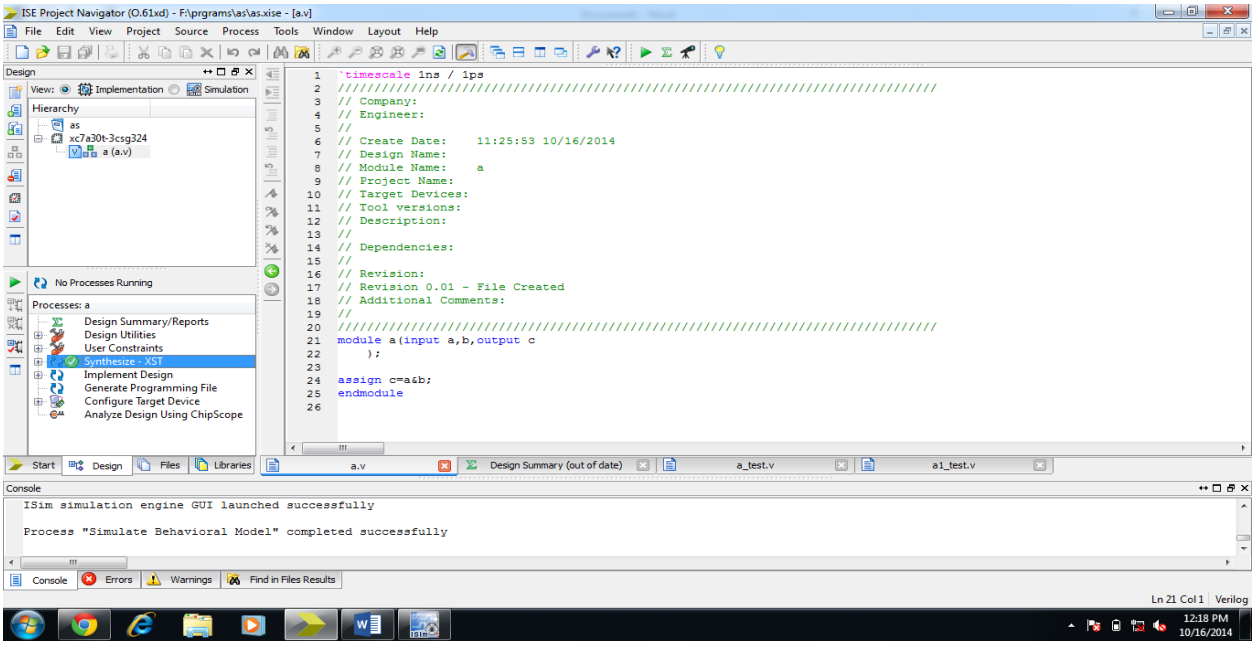
7.It will again displays a dilagbox by fiving details of filename etc, click on next



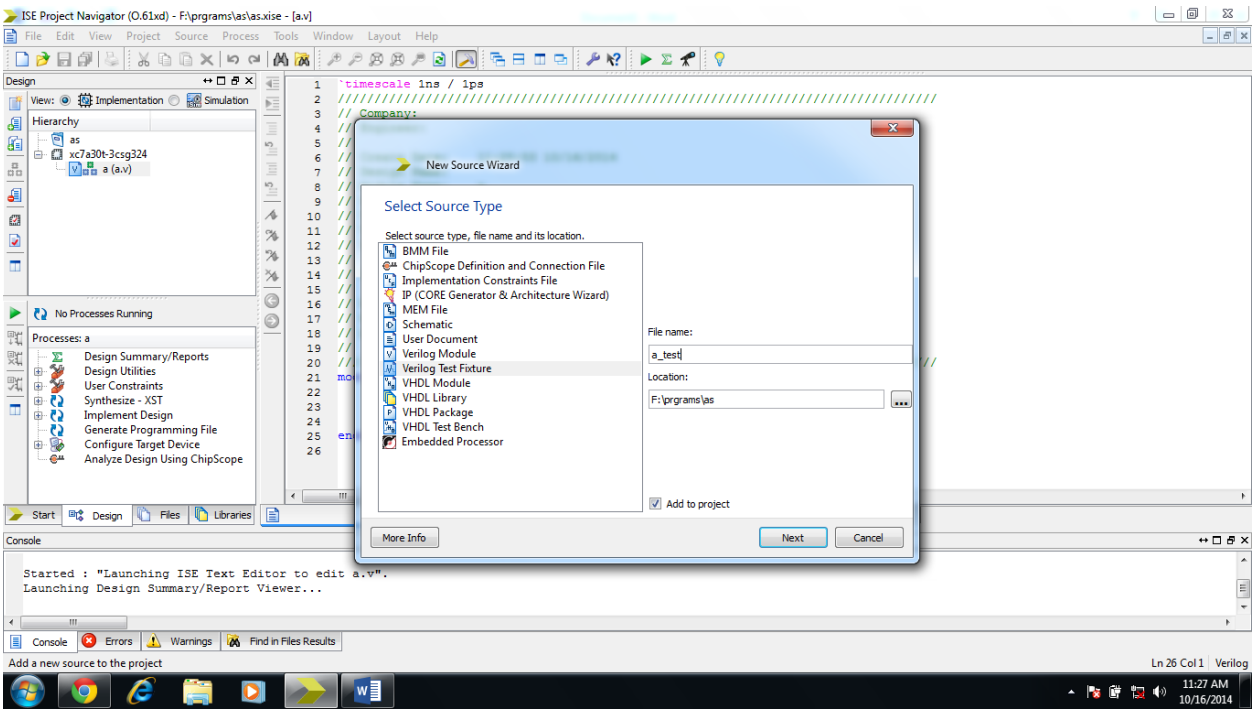
8.It will open a white space in the project window containing filename the double click on the file name so that it will displays respective file window ,where we should write the code



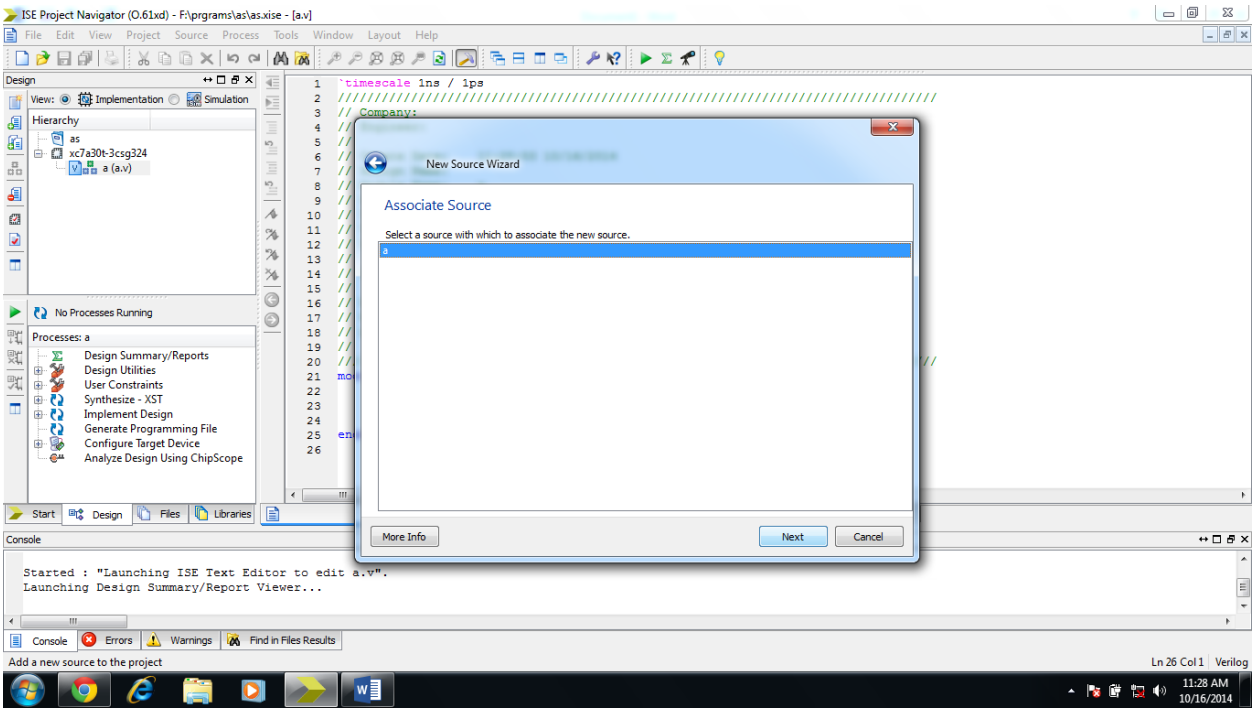
9.After completion writing code select the file name and click on synthesis which will check for errors, if there are any errors in syntax or design errors are checked and shown in the below



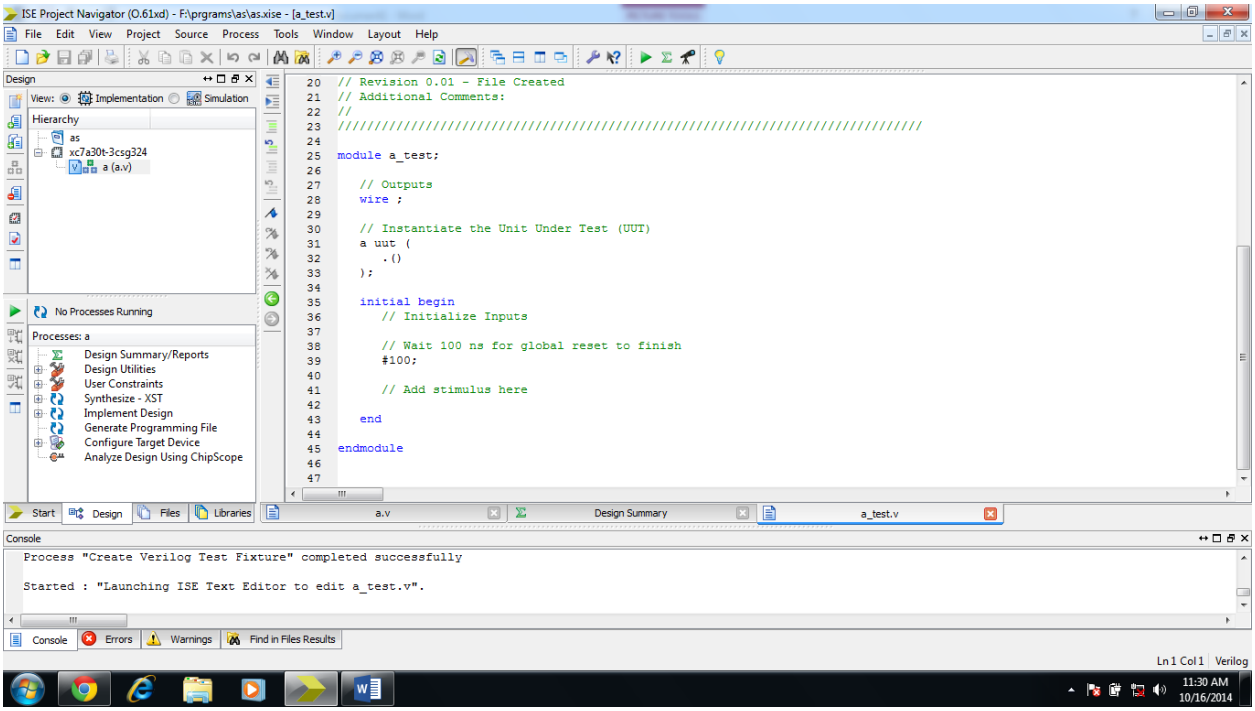
10.After sucessful synthesis we should have to create tesh bench file with extension as test,for that again riht click on the file name as shown below,give filename



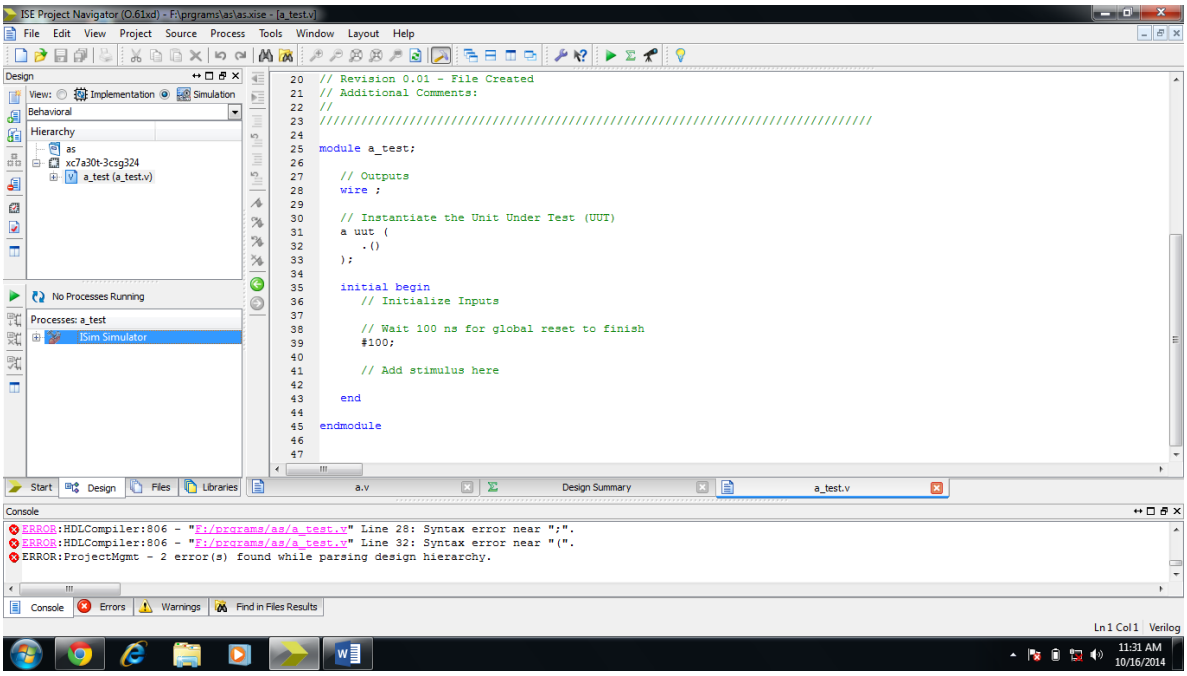
11.If there are list files then select file for which we are creating the test bench. Click on next



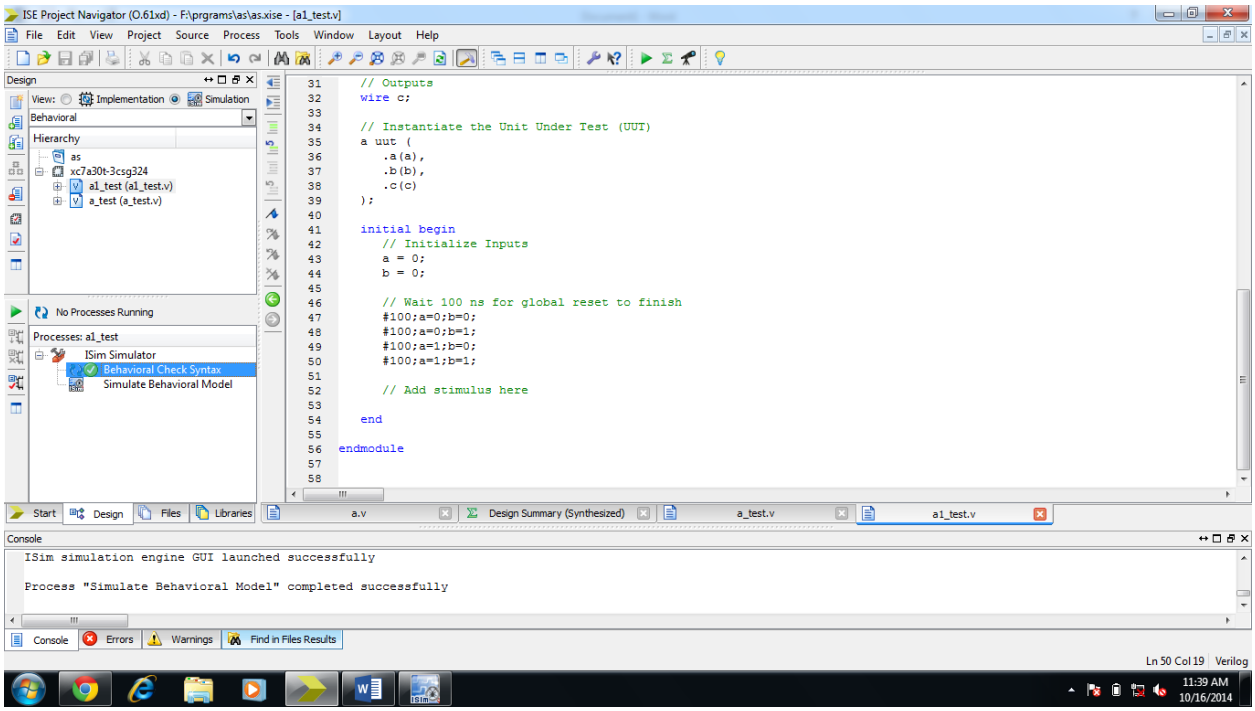
12.It again gives a testbench file in the project window, then give reqired inputs



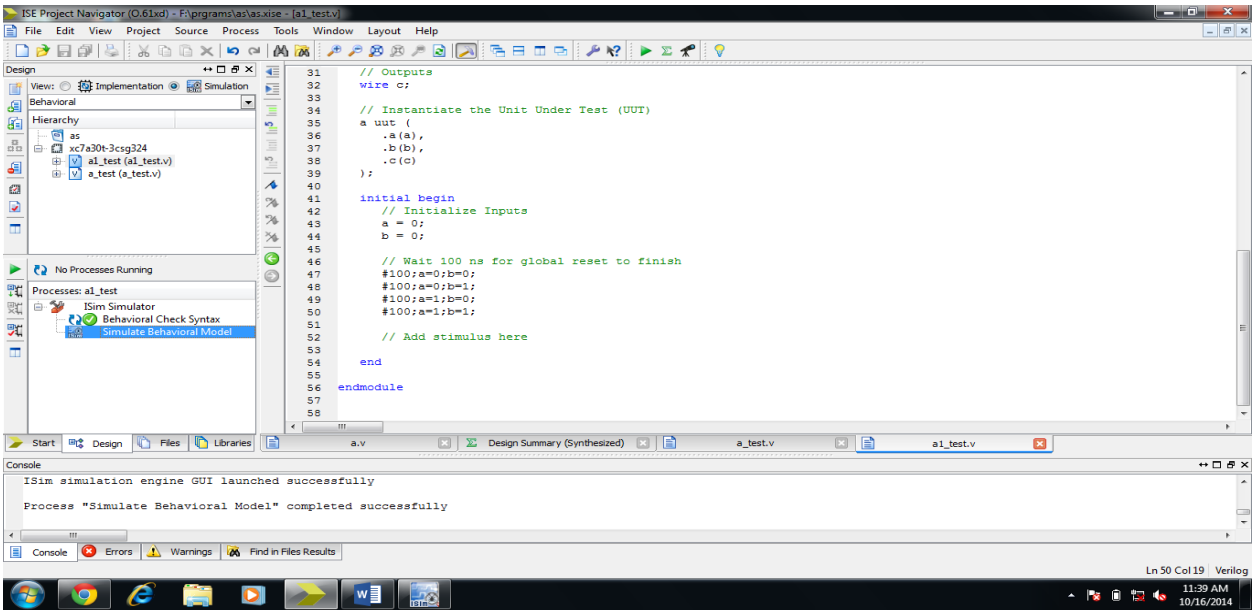
13.select simulation from the view bar in the project window above the hiearchy window as follows.



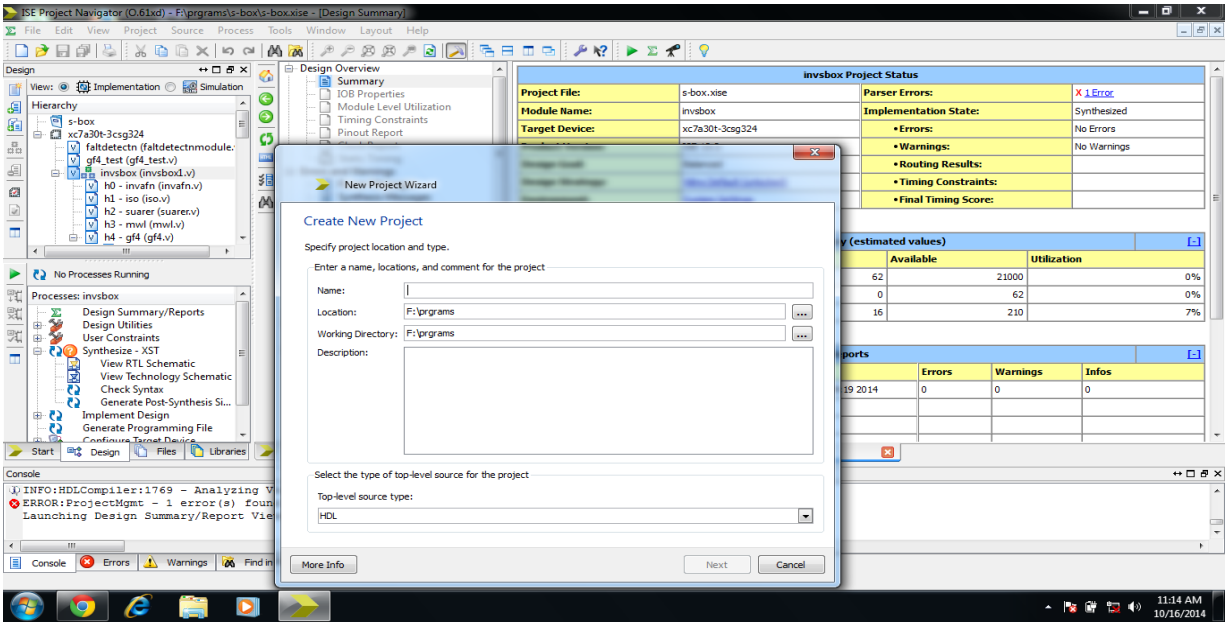
14. Double click on Isim Simulator it will expand as follows click on behavioral check syntax



15. click on simulate behavioral model, it will displays wave form for in response to the inputs given in the test bench file



16.That wave form window having option to zoom out, zoom in to analyze the wave form clearly in order to understand behavior of design



REFERENCES

[1] Yamini R, Ramya M V, Design and Verification of UART using System Verilog, International Journal of Engineering and Advanced Technology (IJEAT)ISSN: 2249 – 8958 (Online), Volume-9 Issue-5, June 2020.

[2] Lakshmi Manasa Kappaganthu, Durga Prakash M, Avinash Yadlapati, I2C Protocol and its Clock Stretching Verification using System Verilog and UVM, International Conference on Inventive Communication and Computational Technologies, (ICICCT 2017).

[1] R. K. Megalingam, J. M. Varghese and S. A. Anil, "Distance estimation and direction finding using I2Cprotocol for an auto-navigation platform," 2016 International Conference on VLSI Systems, Architectures, Technology and Applications (VLSI- SATA), Bengaluru, India, 2016, pp. 1- 4, doi: 10.1109/VLSI-SATA.2016.7593061.

[2] U. Nanda and S. K. Patnaik, "Universal Asynchronous Receiver and Transmitter (UART)," 2016 3rd International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India,2016, pp. 1-5, doi: 10.1109/ICACCS.2016.7586376.

[5] Vibhu Chinmay, Shubham Sachdeva, “A Review Paperon Design and Simulation of UART

for Serial Communication,”2014 IJIRT, Volume 1 Issue 6, SSN : 2349-6002.

[6] Bidisha Kashyap and V Ravi, “Universal Verification Methodology Based Verification of UART Protocol”, Published under licence by IOP Publishing Ltd. Journal of Physics: Conference Series, Volume 1716, National Science, Engineering and Technology Conference (NCSET)2020,11-12 May 2020, Vellore Institute of Technology, Chennai, India. Citation Bidisha Kashyap and V Ravi 2020 J. Phys.: Conf. Ser. 1716 012040, DOI 10.1088/1742-6596/1716/1/012040.

[7] M. R. L. P. M. M. “Design and Implementation of UART”. International Journal on Recent and Innovation Trends in Computing and Communication, vol. 3, no. 6,June 2015, pp. 4289-91, doi:10.17762/ijritcc.v3i6.4638.

[8] Frederic Leens, An Introduction to I2C and SPI Protocols, IEEE Instrumentation and Measurement magazine February 2009.“Implementing I2C Communication Protocol in LABVIEW FPGA”, http:/ /www.ni.com. Samir Palnitkar, Verilog HDL - A guide.

[9] Comparison of VHDL, Verilog and System Verilog, by Stephen Bailey, www.model.com C. H. Roth, “Digital System Design Using VHDL", PWS.

[10] Publishing Company, 2008.

[11] System Verilog Verification guide (https://verificationguide.com/systemverilog/systemverilog-tutorial/).