**A Mini Project**

on

**Operation of Vending Machine for Subsidization**

**Project report Submitted in Partial Fulfilment for the award of Degree of**

**BACHELOR OF TECHNOLOGY**

**IN**

**ELECTRONICS AND COMMUNICATION ENGINEERING**

**SUBMITTED**

**By**

**V.SAI KUMAR 14951A0490**

**B.HEMA 14951A0471**

**R.BHAVANA 14951A0466**

**G.MANIKANTA 15955A0414**

**Under the Supervision of**

**Mr. S Rambabu**

Assistant professor, ECE



**Department of Electronics and Communication Engineering**

**INSTITUTE OF AERONAUTICAL ENGINEERING**

**(Autonomous)**

DUNIDIGAL- 500043, HYDERABAD, TELANGANA STATE

**(Affiliated to JNTU, Hyderabad)**

**2017-2018**

|  |  |  |
| --- | --- | --- |
|  | **INSTITUTE OF AERONAUTICAL ENGINEERING**  DUNDIGAL-500043, HYDERABAD, TELANGANA |  |

**2017-2018**

**Department of Electronics and Communication Engineering**

**CERTIFICATE**

This is to certify that the work embodies in this dissertation entitled, **“OPERATION OF VENDING MACHINE FOR SUBSIDIZATION”** being submitted by **V.SaiKumar (14951A0490), B.Hema (14951A0471), R.Bhavana (14951A0466) and G.Manikanta(15955A0414),** for partial fulfilment of the requirement for the award of ‘**Bachelor of Technology** in **Electronics and Communication Engineering** discipline to Institute of Aeronautical Engineering, Dundigal, Hyderabad during the academic year 2017 - 2018 is a record of bonafide piece of work, undertaken by them in the supervision of the undersigned**.**

**Approved and Supervised by**

**S.RAMBABU**

Electronics and Communication Engineering,

Assistant Professor

**Forwarded by**

**Dr. L V Narasimha Prasad Prof.V.R.Seshagiri Rao Principal Head of Department, ECE**

**External Examiner**

** ELECTRONICS AND COMMUNICATION**

**ENGINEERING**

**INSTITUTE OF AERONAUTICAL ENGINEERING**

**(Autonomous)**

Dundigal, Hyderabad - 500053, Telangana

**2017-2018**

**Department of Electronics and Communication Engineering**

**DECLARATION**

We, the students of “**Bachelor of Technology in Electronics and Communication Engineering Branch”, session: 2017 -2018,** Institute of Aeronautical Engineering, Dundigal, Hyderabad, hereby declare that the work presented in this Project Work entitled “**OPERATION OF VENDING MACHINE FOR SUBSIDIZATION”** is the outcome of our own bonafide work and is correct to the best of our knowledge and this work has been undertaken taking care of Engineering Ethics. It contains no material previously published or written by another person nor material which has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

**Project Associates:**

**V.SaiKumar**

**B.Hema**

**R.Bhavana**

**G.Manikanta**

**ACKNOWLEDGEMENT**

We are greatly indebted to my project guide, **S.Rambabu**, Assistant Professor, Department of Electronics and Communication Engineering, for his invaluable guidance and inspiration which have sustained us to accomplish our work successfully.

We extend our deep gratitude to **S.Rambabu**, Assistant Professor, for his directions, constructive suggestions and continuous support throughout our project work. It helped us as a source of encouragement and support all through the project work.

We have a great pleasure in expressing our sincere thanks to **Prof. V.R.Seshagiri Rao,** Head of the department, who ignited our hidden potential, built career, inculcated self-confidence, sincerity and discipline within us and gave a path of success.

It is our pleasure to acknowledge gratefully to the **Management and Principal**, for their inspiration, valuable suggestions and keen interest during our work.

We are grateful to the teaching and non- teaching faculty members of the Department of Electronics and Communication Engineering, for their encouragement and the facilities provided during our project work.

We appreciate the arduous tasks of our **friends, near** and **dear** who injected patience, fortitude to overcome the challenges that have come our way.

We extend our whole hearted thanks to those who worked above and beyond this long-life time journey.

**ABSTRACT**

A vending machine is a device which automatically dispenses items such as snacks , beverages etc when credit/cash is given as input to it. Crediting to it's high quality fast food processing , It is now one of the most preferred machines in countries like Japan , Singapore and Malaysia.

This paper describes about Designing of a  vending machine with Automated Billing features , It follows the basic principle of Vending machine , Which accepts the Credit/Cash as input and delivers the desired product and the remaining cash as output.

But the main Objective of this paper is that the Vending machine has customized input of barcode scanning, i.e it is grouped for the Subsidization of the items to be sold.

The verilog Code for the above model has been successfully developed and simulated using Xilinx ISE 14.2 software.

The Applications for the above model can be shown as :

* It can be implemented in many places especially colleges so that it can accordingly be useful to students.
* More experienced people in an organization can have better subsidized rates compared to others with a grouped barcode.
* As it is accessed by a barcode, the operations can be secured.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **CONTENTS** | | | | | | |
| Chapter | Description | | | | | Page No. |
| 1 | **INTRODUCTION** | | | | | 1 |
|  | 1.1 | Early Days | | | | 1 |
|  | 1.2 | Growth | | | | 1 |
|  | 1.3 | Today | | | | 2 |
|  | 1.4 | Technology | | | | 3 |
|  | 1.5 | Modern Vending Machines | | | | 5 |
|  | 1.6 | Mechanism | | | | 6 |
| 2 | **XILINX SOFTWARE** | | | | |  |
|  | 2.1 | Introduction | | | | 7 |
|  | 2.2 | Programmable Logic Device | | | | 8 |
|  | 2.3 | Creating a new project | | | | 8 |
|  |  | 2.3.1 | | Opening a Project | | 8 |
|  |  | 2.3.2 | | Project Name | | 9 |
|  |  | 2.3.3 | | Project Location | | 9 |
|  | 2.4 | Device and Design Flow of Project | | | | 10 |
|  | 2.5 | Creating a Verilog HDL input file for a combinational logic | | | | 11 |
| 3 | **TEST BENCH** | | | | |  |
|  | 3.1 | Creation | | | | 14 |
|  | 3.2 | Select the Family, Device, Package and Speed | | | | 15 |
|  | 3.3 | Define Module | | | | 16 |
|  | 3.4 | Generating a Self-Checking Test Bench | | | | 17 |
|  | 3.5 | To generate a Self-Checking Test Bench | | | | 18 |
| 4 | **STATE DIAGRAMS** | | | | |  |
|  | 4.1 | First Year | | | | 21 |
|  |  | 4.1.1 | | Cancellation | | 21 |
|  | 4.2 | Second Year | | | | 22 |
|  |  | 4.2.1 | | Cancellations | | 22 |
|  | 4.3 | Third Year | | | | 23 |
|  |  | 4.3.1 | | Cancellation | | 23 |
|  | 4.4 | Fourth Year | | | | 24 |
| 5 | **RESULTS** | | | | |  |
|  | 5.1 | | First Year | | | 25 |
|  |  | | 5.1.1 | | Timings | 26 |
|  | 5.2 | | Second Year | | | 27 |
|  | 5.3 | | Third Year | | | 28 |
|  | 5.4 | | Fourth Year | | | 29 |
|  | 5.5 | | Cancellation | | | 30 |
|  |  | | 5.5.1 | | First Year | 30 |
|  |  | | 5.5.2 | | Second Year | 32 |
|  |  | | 5.5.3 | | Third Year | 33 |
|  |  | | 5.5.4 | | Fourth Year | 33 |
| 6 | **CONCLUSION** | | | | | 34 |
| 7 | **REFERENCES** | | | | | 35 |

**CHAPTER -1**

**INTRODUCTION**

**1.1 EARLY DAYS**

[Ross Freeman](https://en.wikipedia.org/wiki/Ross_Freeman), [Bernard Vonderschmitt](https://en.wikipedia.org/wiki/Bernard_Vonderschmitt), and James V Barnett II, who all had worked for [integrated circuit](https://en.wikipedia.org/wiki/Integrated_circuit) and solid-state device manufacturer [Zilog](https://en.wikipedia.org/wiki/Zilog) Corp, founded Xilinx in 1984.

While working for Zilog, Freeman wanted to create chips that acted like a blank tape, allowing users to program the technology themselves. At the time, the concept was [paradigm](https://en.wikipedia.org/wiki/Paradigm) changing. "The concept required lots of [transistors](https://en.wikipedia.org/wiki/Transistors) and, at that time, transistors were considered extremely precious people thought that Ross's idea was pretty far out", said Xilinx Fellow Bill Carter, who when hired in 1984, as the first IC designer, was Xilinx's eighth employee.

Big semiconductor manufacturers were enjoying strong profits by producing massive volumes of generic circuits. Designing and manufacturing dozens of different circuits for specific markets offered lower profit margins and required greater manufacturing complexity. What became known as the [FPGA](https://en.wikipedia.org/wiki/Field-programmable_gate_array) would allow circuits produced in quantity to be tailored by individual market segments.

Freeman failed to convince Zilog to invest in creating the FPGA to chase what was only a $100 million market at the time. Freeman and Barnett left Zilog and teamed up with their 60-year-old ex-colleague Bernard Vonderschmitt to raise $4.5 million in [venture](https://en.wikipedia.org/wiki/Venture_capital) [funding](https://en.wikipedia.org/wiki/Funding) to design the first commercially viable FPGA. They incorporated the company in 1984 and began selling its first product by 1985.

By late 1987 the company had raised more than $18 million in [venture capital](https://en.wikipedia.org/wiki/Venture_capital) (equivalent to $37.95 million in 2016) and generated revenues at an annualized rate of nearly $14 million.

**1.2 GROWTH**

As demand for programmable logic continued to grow, so did Xilinx's revenues and profits.

From 1988 to 1990, the company's revenue grew each year from $30 million to $50 million to $100 million. During this time period, the company which had been providing funding to Xilinx, Monolithic Memories Inc. (MMI), was purchased by Xilinx competitor [AMD](https://en.wikipedia.org/wiki/AMD). As a result, Xilinx dissolved the deal with MMI and went public on the [NASDAQ](https://en.wikipedia.org/wiki/NASDAQ) in 1989.[[6]](https://en.wikipedia.org/wiki/Xilinx#cite_note-four-6) The company also moved to a 144,000-square-foot (13,400 m2) plant in San Jose, California in order to keep pace with demand from companies like [HP](https://en.wikipedia.org/wiki/Hewlett-Packard), [Apple Inc.](https://en.wikipedia.org/wiki/Apple_Inc.), [IBM](https://en.wikipedia.org/wiki/IBM) and [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems) who were buying large quantities from Xilinx.

Xilinx's sales rose from $560 million in 1996 to $2.2 billion by the end of its fiscal year 2013. Moshe Gavrielov  an [EDA](https://en.wikipedia.org/wiki/Electronic_design_automation) and [ASIC](https://en.wikipedia.org/wiki/Application-specific_integrated_circuit) industry veteran who was appointed as president and CEO in early 2008 – introduced targeted design platforms that combine FPGAs with [software](https://en.wikipedia.org/wiki/Software), IP cores, boards and kits to address focused target applications.[[17]](https://en.wikipedia.org/wiki/Xilinx#cite_note-embedded-17) These targeted design platforms are an alternative to costly application-specific integrated circuits ([ASICs](https://en.wikipedia.org/wiki/ASICs)) and application-specific standard products (ASSPs).

**1.3 TODAY**

The company has expanded its product portfolio since its founding. Xilinx sells a broad range of FPGAs, [complex programmable logic devices](https://en.wikipedia.org/wiki/Complex_programmable_logic_device) (CPLDs), design tools, intellectual property and reference designs. Xilinx also has a global services and training program.

After using the introduction of 3D chips to deliver more powerful FPGAs, Xilinx then adapted the technology to combine formerly separate components in a single chip, first combining an FPGA with transceivers to boost bandwidth capacity while using less power. According to Xilinx CEO Moshe Gavrielov, the addition of a heterogeneous communications device, combined with the introduction of new software tools and the Zynq-7000 line of 28 nm SoC devices that combine an [ARM core](https://en.wikipedia.org/wiki/ARM_core) with an FPGA, are part of shifting its position from a programmable logic device supplier to one delivering “all things programmable”.

The company's products have been recognized by EE Times, EDN and others for innovation and market impact.

In addition to Zynq-7000, Xilinx product lines (see Current Family Lines) include the [Virtex](https://en.wikipedia.org/wiki/Virtex_(FPGA)), Kintex and Artix series, each including configurations and models optimized for different applications. With the introduction of the Xilinx 7 series in June, 2010, the company has moved to three major FPGA product families, the high-end [Virtex](https://en.wikipedia.org/wiki/Virtex_(FPGA)), the mid-range Kintex family and the low-cost Artix family, retiring the Spartan brand, which ends with the Xilinx Series 6 FPGAs. In April 2012, the company introduced the [Vivado Design Suite](https://en.wikipedia.org/wiki/Xilinx_Vivado) - a next-generation SoC-strength design environment for advanced electronic system designs. In May, 2014, the company shipped the first of the next generation FPGAs: the 20 nm UltraScale.

**1.4 TECHNOLOGY**

Xilinx designs, develops and markets programmable logic products, including integrated circuits (ICs), software design tools, predefined system functions delivered as intellectual property (IP) cores, design services, customer training, field engineering and technical support. Xilinx sells both FPGAs and CPLDs for electronic equipment manufacturers in end markets such as [communications](https://en.wikipedia.org/wiki/Communications),mechanical, [industrial](https://en.wikipedia.org/wiki/Industry), [consumer](https://en.wikipedia.org/wiki/Consumer), [automotive](https://en.wikipedia.org/wiki/Automotive) and [data processing](https://en.wikipedia.org/wiki/Data_processing).

Xilinx's FPGAs have been used for the [ALICE](https://en.wikipedia.org/wiki/A_Large_Ion_Collider_Experiment) (A Large Ion Collider Experiment) at the [CERN](https://en.wikipedia.org/wiki/CERN) European laboratory on the [French](https://en.wikipedia.org/wiki/France)-[Swiss](https://en.wikipedia.org/wiki/Switzerland) border to map and disentangle the trajectories of thousands of [subatomic particles](https://en.wikipedia.org/wiki/Subatomic_particles). Xilinx has also engaged in a partnership with the United States Air Force Research Laboratory’s Space Vehicles Directorate to develop FPGAs to withstand the damaging effects of radiation in space, which are 1,000 times less sensitive to space radiation than the commercial equivalent, for deployment in new satellites.

The Virtex-II Pro, Virtex-4, Virtex-5, and Virtex-6 FPGA families, which include up to two embedded IBM PowerPC cores, are targeted to the needs of [system-on-chip](https://en.wikipedia.org/wiki/System-on-chip) (SoC) designers.

Xilinx FPGAs can run a regular embedded OS (such as [Linux](https://en.wikipedia.org/wiki/Linux) or [vxWorks](https://en.wikipedia.org/wiki/VxWorks)) and can implement processor peripherals in programmable logic.

Xilinx's IP cores include IP for simple functions ([BCD](https://en.wikipedia.org/wiki/Binary-coded_decimal) encoders, counters, etc.), for domain specific cores ([digital signal processing](https://en.wikipedia.org/wiki/Digital_signal_processing), [FFT](https://en.wikipedia.org/wiki/Fast_Fourier_transform) and [FIR](https://en.wikipedia.org/wiki/Free_ideal_ring) cores) to complex systems (multi-gigabit networking cores, the MicroBlaze soft microprocessor and the compact Picoblaze microcontroller). Xilinx also creates custom cores for a fee.

The main design toolkit Xilinx provides engineers is the [Vivado Design Suite](https://en.wikipedia.org/wiki/Xilinx_Vivado), an integrated design environment (IDE) with a system-to-IC level tools built on a shared scalable data model and a common debug environment. Vivado includes electronic system level (ESL) design tools for synthesizing and verifying C-based algorithmic IP; standards based packaging of both algorithmic and RTL IP for reuse; standards based IP stitching and systems integration of all types of system building blocks; and the verification of blocks and systems. A free version WebPACK Edition of Vivado provides designers with a limited version of the design environment.

Xilinx's Embedded Developer's Kit (EDK) supports the embedded [PowerPC](https://en.wikipedia.org/wiki/PowerPC) 405 and 440 cores (in Virtex-II Pro and some Virtex-4 and -5 chips) and the [Microblaze](https://en.wikipedia.org/wiki/Microblaze) core. Xilinx's System Generator for DSP implements DSP designs on Xilinx FPGAs. A freeware version of its EDA software called ISE WebPACK is used with some of its non-high-performance chips. Xilinx is the only (as of 2007) FPGA vendor to distribute a native Linux freeware synthesis tool chain.

Xilinx announced the architecture for a new [ARM Cortex-A9](https://en.wikipedia.org/wiki/ARM_Cortex-A9)-based platform for embedded systems an designer that combines the software programmability of an embedded processor with the hardware flexibility of an FPGA. The new architecture abstracts much of the hardware burden away from the embedded software developers' point of view, giving them an unprecedented level of control in the development process. With this platform, software developers can leverage their existing system code based on ARM technology and utilize vast off-the-shelf open-source and commercially available software component libraries. Because the system boots an OS at reset, software development can get under way quickly within familiar development and debug environments using tools such as ARM's Real View development suite and related third-party tools, Eclipse-based IDEs, GNU, the Xilinx Software Development Kit and others. In early 2011, Xilinx began shipping a new device family based on this architecture. The Zynq-7000 SoC platform immerses ARM multi-cores, programmable logic fabric, DSP data paths, memories and I/O functions in a dense and configurable mesh of interconnect. The platform targets embedded designers working on market applications that require multi-functionality and real-time responsiveness, such as automotive driver assistance, intelligent video surveillance, industrial automation, aerospace and defense, and next-generation wireless.

Following the introduction of its 28 nm 7-series FPGAs, Xilinx revealed that several of the highest-density parts in those FPGA product lines will be constructed using multiple dies in one package, employing technology developed for 3D construction and stacked-die assemblies. The company’s stacked silicon interconnect (SSI) technology stacks several (three or four) active FPGA dies side-by-side on a silicon [interposer](https://en.wikipedia.org/wiki/Interposer) – a single piece of silicon that carries passive interconnect. The individual FPGA dies are conventional, and are flip-chip mounted by microbumps on to the interposer. The interposer provides direct interconnect between the FPGA dies, with no need for transceiver technologies such as high-speed SERDES. In October 2011, Xilinx shipped the first FPGA to use the new technology, the Virtex-7 2000T FPGA, which includes 6.8 billion transistors and 20 million ASIC gates. The following spring, Xilinx used the 3D technology to ship the Virtex-7 HT, the industry’s first heterogeneous FPGAs, which combine high bandwidth FPGAs with up to sixteen 28 Gbit/s and seventy-two 13.1 Gbit/s transceivers to reduce power and size requirements for key Nx100G and 400G line card applications and functions.

In January 2011, Xilinx acquired design tool firm Auto ESL Design Technologies and added System C high-level design for its 6- and 7-series FPGA families. The addition of Auto ESL tools extends the design community for FPGAs to designers more accustomed to designing at a higher level of abstraction using C, C++ and System C.

In April 2012, Xilinx introduced a redesign of its toolset for programmable systems, called [Vivado Design Suite](https://en.wikipedia.org/wiki/Xilinx_Vivado). This IP and system-centric design software supports newer high capacity devices, and speeds the design of programmable logic and I/O. Vivado provides faster integration and implementation for programmable systems into devices with 3D stacked silicon interconnect technology, ARM processing systems, analog mixed signal (AMS), and many semiconductor intellectual property (IP) cores.

The earliest known reference to a vending machine is in the work of [Hero of Alexandria](https://en.wikipedia.org/wiki/Hero_of_Alexandria), an engineer and mathematician in first-century [Roman Egypt](https://en.wikipedia.org/wiki/Roman_Egypt). His machine accepted a coin and then dispensed [holy water](https://en.wikipedia.org/wiki/Holy_water). When the coin was deposited, it fell upon a pan attached to a lever. The lever opened a valve which let some water flow out. The pan continued to tilt with the weight of the coin until it fell off, at which point a counterweight snapped the lever up and turned off the valve.

Coin-operated machines that dispensed tobacco were being operated as early as 1615 in the [taverns](https://en.wikipedia.org/wiki/Tavern) of England. The machines were portable and made of [brass](https://en.wikipedia.org/wiki/Brass). An English bookseller, Richard Carlile, devised a newspaper dispensing machine for the dissemination of banned works in 1822. Simeon Denham was awarded British Patent no. 706 for his stamp dispensing machine in 1867, the first fully automatic vending machine.

### 1.5 MODERN VENDING MACHINES

[](https://en.wikipedia.org/wiki/File:Automatic_Stamp_and_Postcard_Vending_Machine.jpg)

The first modern coin-operated vending machines were introduced in [London](https://en.wikipedia.org/wiki/London), [England](https://en.wikipedia.org/wiki/England) in the early 1880s, dispensing [postcards](https://en.wikipedia.org/wiki/Postcard). The machine was invented by Percival Everitt in 1883 and soon became a widespread feature at railway stations and post offices, dispensing [envelopes](https://en.wikipedia.org/wiki/Envelopes), [postcards](https://en.wikipedia.org/wiki/Postcard), and [notepaper](https://en.wikipedia.org/wiki/Notepaper). The Sweetmeat Automatic Delivery Company was founded in 1887 in England as the first company to deal primarily with the installation and maintenance of vending machines. In 1893, [Stollwerck](https://en.wikipedia.org/wiki/Stollwerck), a German chocolate manufacturer, was selling its chocolate in 15,000 vending machines. It set up separate companies in various territories to manufacture vending machines to sell not just chocolate, but cigarettes, matches, chewing gum and soap products.

The first vending machine in the U.S. was built in 1888 by the [Thomas Adams Gum Company](https://en.wikipedia.org/wiki/Thomas_Adams_Gum_Company), selling gum on New York City train platforms. The idea of adding games to these machines as a further incentive to buy came in 1897 when the Pulver Manufacturing Company added small figures, which would move around whenever somebody bought some gum from their machines. This idea spawned a whole new type of mechanical device known as the "trade stimulators".

## **1.6 MECHANISMS**

After payment has been tendered, a product may become available by:

* the machine releasing it, so that it falls in an open compartment at the bottom, or into a cup, either released first, or put in by the customer, or
* the unlocking of a door, drawer, or turning of a knob.

Some products need to be prepared to become available. For example, tickets are printed or magnetized on the spot, and coffee is freshly concocted. One of the most common form of vending machine, the snack machine, often uses a metal coil which when ordered rotates to release the product.

**CHAPTER 2**

**XILINX SOFTWARE**

**2.1.INTRODUCTION**

Xilinx Tools is a suite of software tools used for the design of digital circuits implemented using Xilinx Field Programmable Gate Array (FPGA) or Complex Programmable Logic Device (CPLD).

The design procedure consists of

(a) design entry,

(b) synthesis and implementation of the design,

(c) functional simulation and

(d) testing and verification.

Digital designs can be entered in various ways using the above CAD tools: using a schematic entry tool, using a hardware description language (HDL) – Verilog or VHDL or a combination of both. In this lab we will only use the design flow that involves the use of Verilog HDL. The CAD tools enable you to design combinational and sequential circuits starting with Verilog HDL design specifications.

The **steps** of this design procedure are listed below:

1. Create Verilog design input file(s) using template driven editor.

2. Compile and implement the Verilog design file(s).

3. Create the test-vectors and simulate the design (functional simulation) without using a PLD (FPGA or CPLD).

4. Assign input/output pins to implement the design on a target device.

5. Download bitstream to an FPGA or CPLD device.

6. Test design on FPGA/CPLD device.

A Verilog input file in the Xilinx software environment consists of the following segments:

**Header**: module name, list of input and output ports.

**Declarations:** input and output ports, registers and wires.

**Logic Descriptions**: equations, state machines and logic functions.

**End**: endmodule

All your designs for this lab must be specified in the above Verilog input format.

Note that the state diagram segment does not exist for combinational logic designs.

**2.2 Programmable Logic Device**

FPGA In this lab digital designs will be implemented in the Basys2 board which has a Xilinx Spartan3E –XC3S250E FPGA with CP132 package. This FPGA part belongs to the Spartan family of FPGAs. These devices come in a variety of packages. We will be using devices that are packaged in 132 pin package with the following part number: XC3S250E-CP132. This FPGA is a device with about 50K gates. Detailed information on this device is available at the Xilinx website.

**2.3 Creating a New Project**

Xilinx Tools can be started by clicking on the Project Navigator Icon on the Windows desktop. This should open up the Project Navigator window on your screen. This window shows (see Figure 1) the last accessed project.

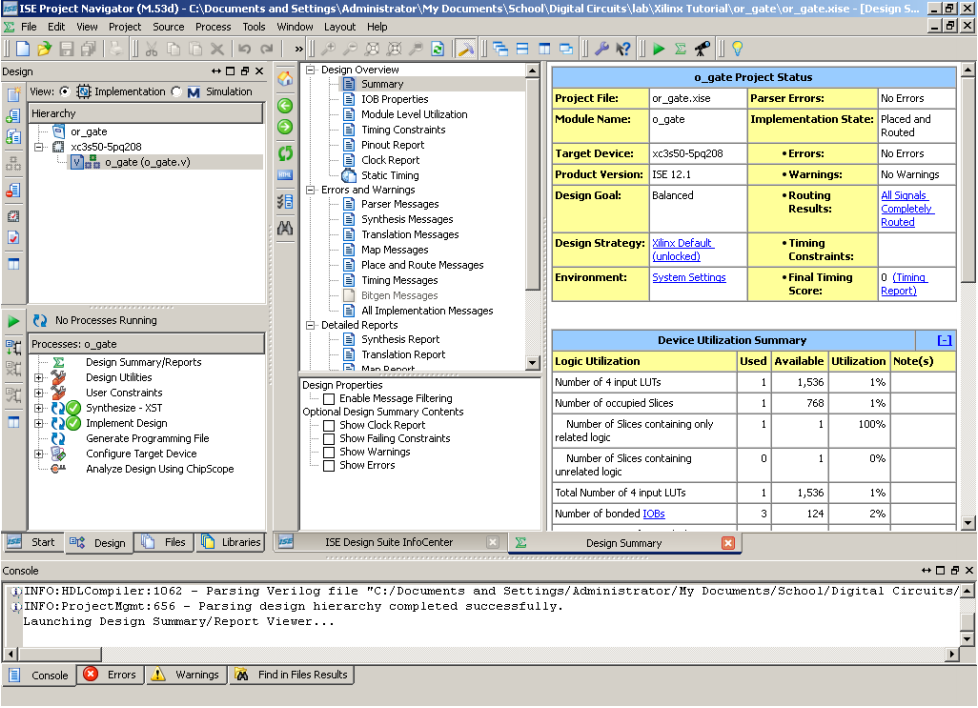


Figure 1: Xilinx Project Navigator window (snapshot from Xilinx ISE software)

**2.3.1 Opening a Project**

Select File->**New Project** to create a new project. This will bring up a new project window (Figure 2) on the desktop. Fill up the necessary entries as follows

**2.3.2 Project Name**: Write the name of your new project

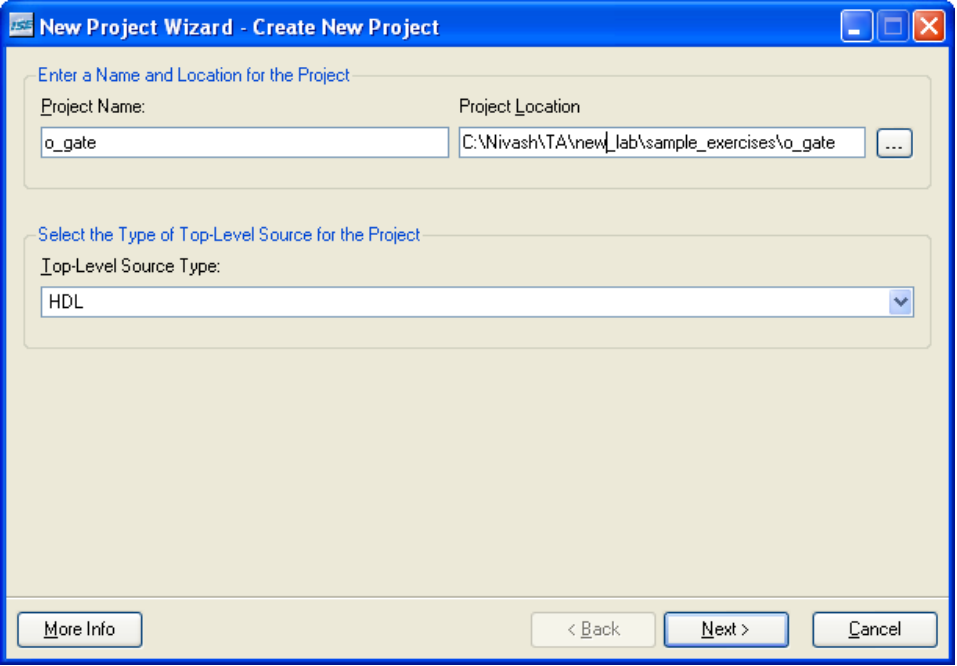


Figure 2: New Project Initiation window (snapshot from Xilinx ISE software)

**2.3.3 Project Location**:

The directory where you want to store the new project (Note: DO NOT specify the project location as a folder on Desktop or a folder in the Xilinx\bin directory. Your H: drive is the best place to put it.

The project location path is NOT to have any spaces in it eg: C:\Nivash\TA\new lab\sample exercises\o\_gate is NOT to be used)

Leave the top level module type as HDL.

Example: If the project name were “o\_gate”, enter “o\_gate” as the project name and then click “Next”.

Clicking on **NEXT** should bring up the following window:

**2.4 Device and Design Flow of Project**

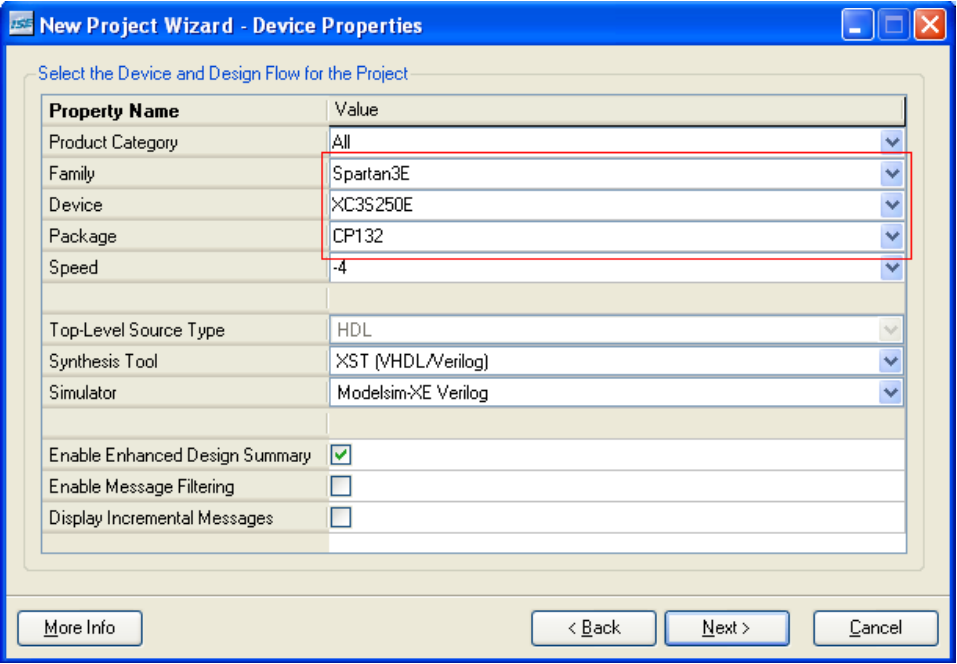


Figure 3: Device and Design Flow of Project (snapshot from Xilinx ISE software) For each of the properties given below, click on the ‘value’ area and select from the list of values that appear.

* **Device Family**: Family of the FPGA/CPLD used. In this laboratory we will be using the Spartan3E FPGA’s
* **Device**: The number of the actual device. For this lab you may enter XC3S250E (this can be found on the attached prototyping board)
* **Package**: The type of package with the number of pins. The Spartan FPGA used in this lab is packaged in CP132 package.
* **Speed** **Grade**: The Speed grade is “-4”
* **Synthesis** **Tool**: XST [VHDL/Verilog].
* **Simulator**: The tool used to simulate and verify the functionality of the design. Modelsim simulator is integrated in the Xilinx ISE. Hence choose “Modelsim-XE Verilog” as the simulator or even Xilinx ISE Simulator can be used.
* Then click on **NEXT** to save the entries.

All project files such as schematics, netlists, Verilog files, VHDL files, etc., will be stored in a subdirectory with the project name. A project can only have one top level HDL source file (or schematic). Modules can be added to the project to create a modular, hierarchical design (see Section 9).

In order to open an existing project in Xilinx Tools, select **File->Open Project** to show the list of projects on the machine. Choose the project you want and click **OK**.

Clicking on **NEXT** on the above window brings up the following window:

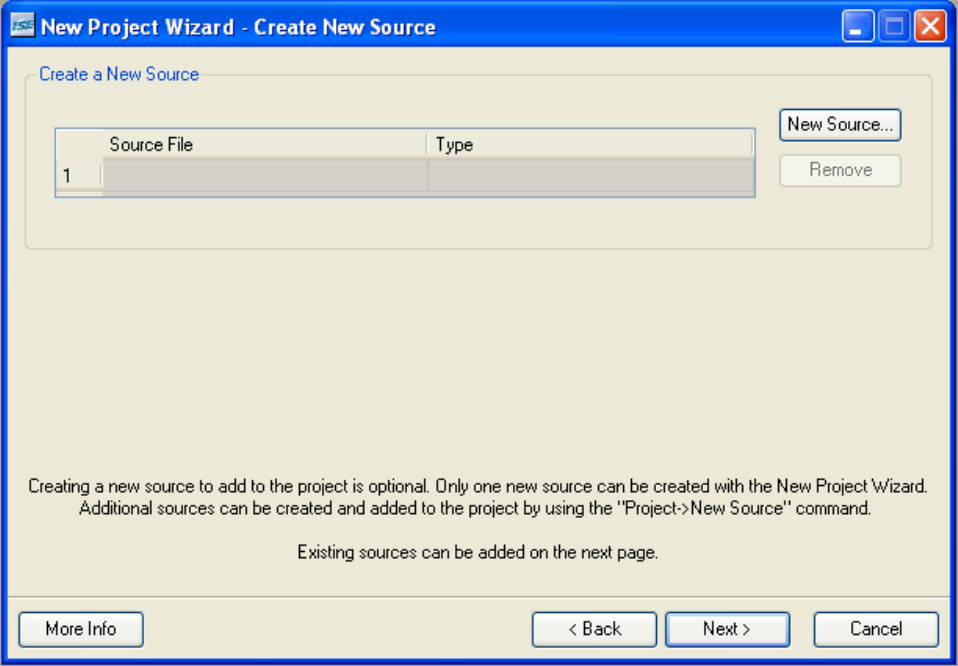


Figure 4: Create New source window (snapshot from Xilinx ISE software)

If creating a new source file, Click on the NEW SOURCE.

**2.5 Creating a Verilog HDL input file for a combinational logic**

Design In this lab we will enter a design using a structural or RTL description using the Verilog HDL. You can create a Verilog HDL input file (.v file) using the HDL Editor available in the Xilinx ISE Tools (or any text editor).

In the previous window, click on the NEW SOURCE

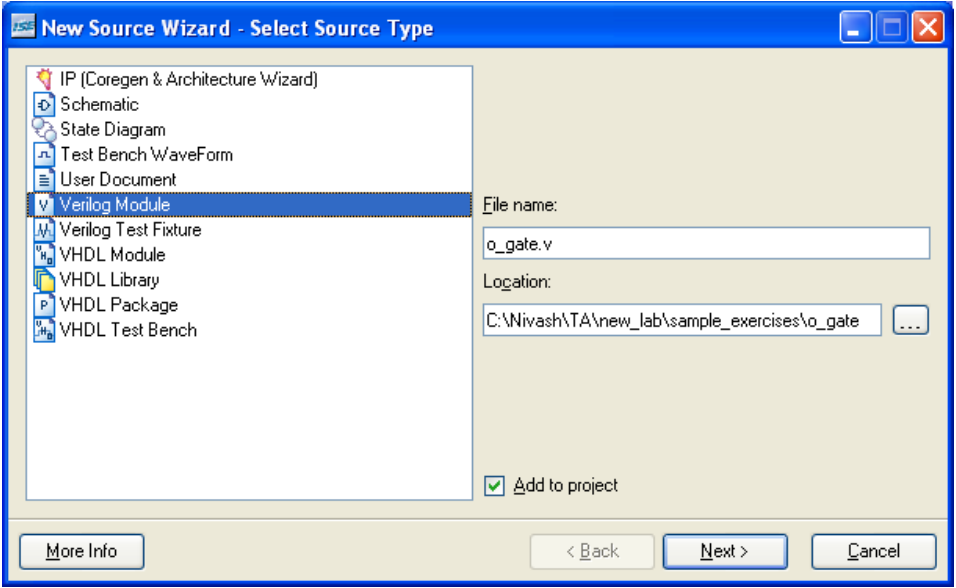
A window pops up as shown in Figure 4. (Note: “Add to project” option is selected by default. If you do not select it then you will have to add the new source file to the project manually.) 

Figure 5: Creating Verilog-HDL source file (snapshot from Xilinx ISE software)

Select Verilog Module and in the “File Name:” area, enter the name of the Verilog source file you are going to create. Also make sure that the option Add to project is selected so that the source need not be added to the project again. Then click on Next to accept the entries. This pops up the following window (Figure 5).

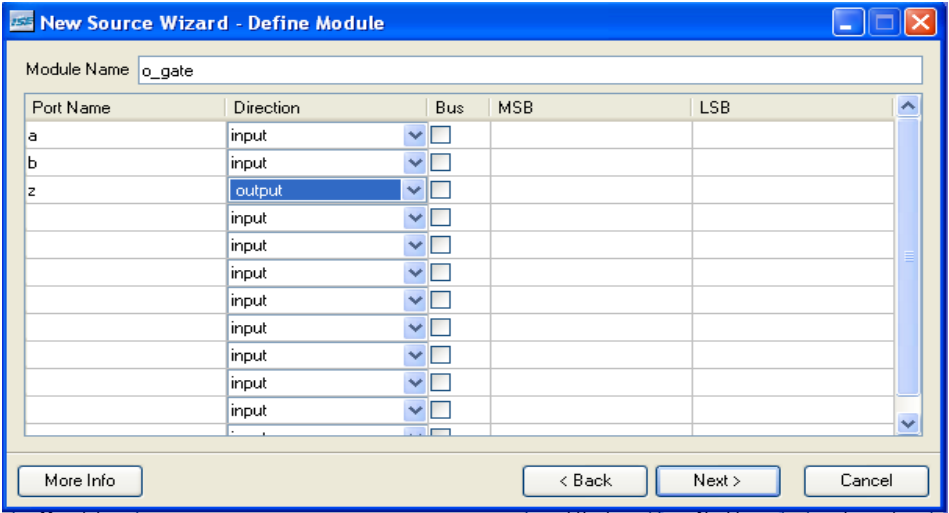


Figure 6: Define Verilog Source window (snapshot from Xilinx ISE software).

In the Port Name column, enter the names of all input and output pins and specify the Direction accordingly. A Vector/Bus can be defined by entering appropriate bit numbers in the MSB/LSB columns. Then click on Next> to get a window showing all the new source information (Figure 6). If any changes are to be made, just click on Next > Next > Finish to continue.

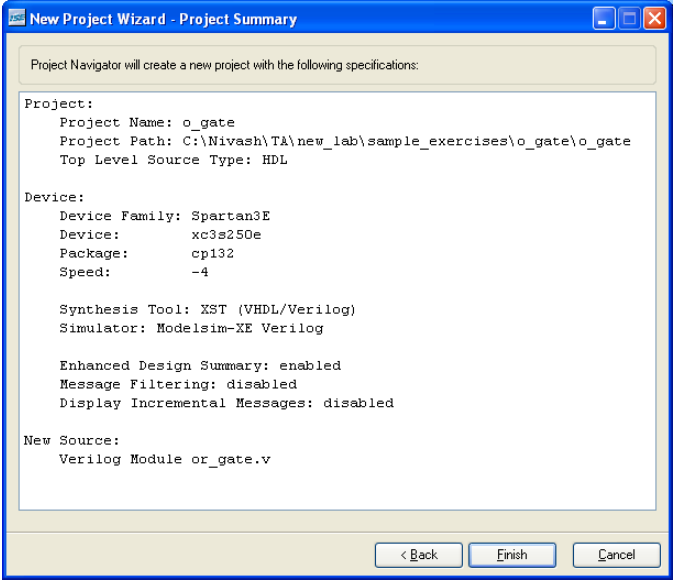


Figure 7: New Project Information window(snapshot from Xilinx ISE software)

Once you click on Finish, the source file will be displayed in the sources window in the Project Navigator (Figure 1). If a source has to be removed, just right click on the source file in the Sources in Project window in the Project Navigator and select Remove in that. Then select Project -> Delete Implementation Data from the Project Navigator menu bar to remove any related files.

**CHAPTER 3**

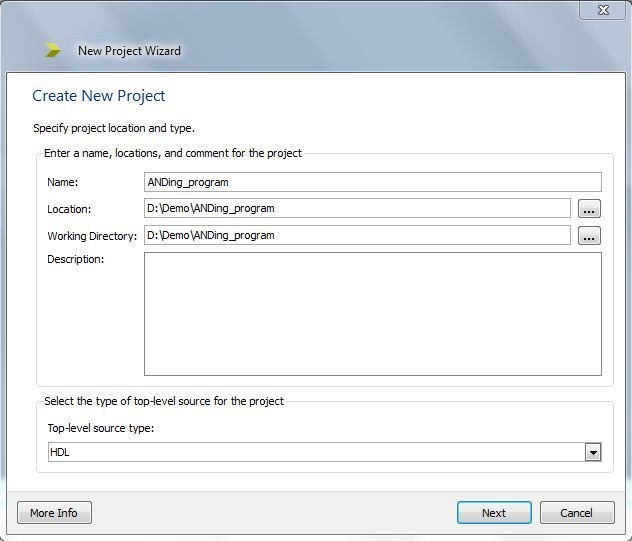
**TEST BENCH**

**3.1 CREATION**

Your design has input signals and output signals, and you need to control the inputs and verify that the outputs are correct. Ideally you want your test to create the input stimulus and verify the correctness of the output signals. In the real world, we want tests to print out what they are doing as the test runs, print any error messages if any of the tests fail, and print a summary about whether the test passed or failed.

The following steps will clearly explains us how to make use of Vivado ISE:

Open Vivado ISE and create a new project.

s

1. Select **Project > New Source**.

**Note**Alternatively, you can double-click **Create New Source** in the [Processes tab](https://www.xilinx.com/itp/xilinx10/isehelp/pn_r_processes_tab.htm).

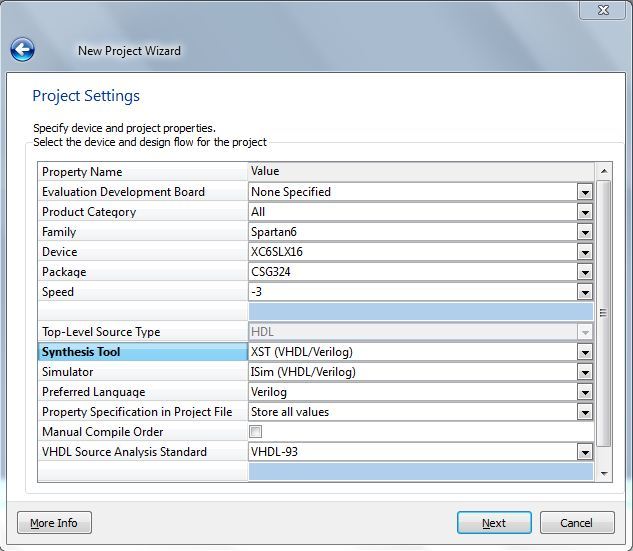
1. In the [New Source Wizard](https://www.xilinx.com/itp/xilinx10/isehelp/pn_db_nsw_select_source_type.htm), select the type of source you want to create.

Different source types are available depending on your project properties (top-level module type, device type, synthesis tool, and language). Some source types launch additional tools to help you create the file, as described in [Source File Types](https://www.xilinx.com/itp/xilinx10/isehelp/ise_r_source_types.htm).

1. Enter a name for the new source file in the File Name field. Follow the naming conventions described in [File Naming Conventions](https://www.xilinx.com/itp/xilinx10/isehelp/ise_r_file_names.htm).
2. In the Location field, enter the directory name or browse to the directory.
3. Select **Add to Project** to automatically add this source to the project.
4. Click **Next**.
5. If you are creating a source file that needs to be associated with an existing source file, select the appropriate source file, and click **Next**. If this does not apply, skip to the next step.
6. In the New Source Wizard - Summary window, verify the information for the new source, and click **Finish**

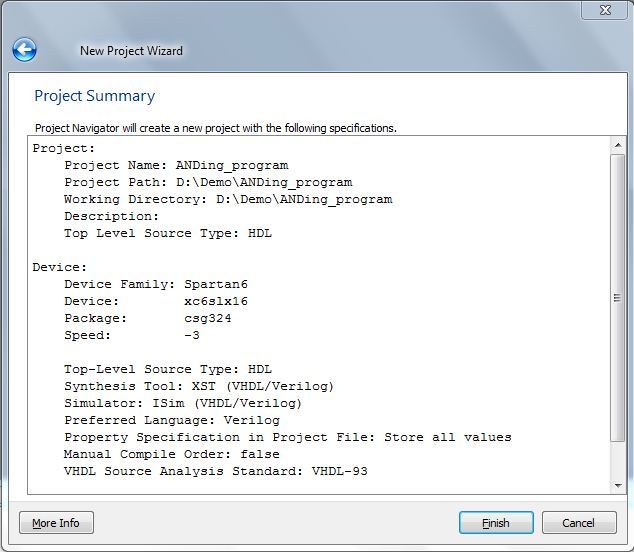
**3.2 SELECT THE FAMILY, DEVICE, PACKAGE AND SPEED**

select your programming language(Verilog/VHDL). Here i am using Verilog language.

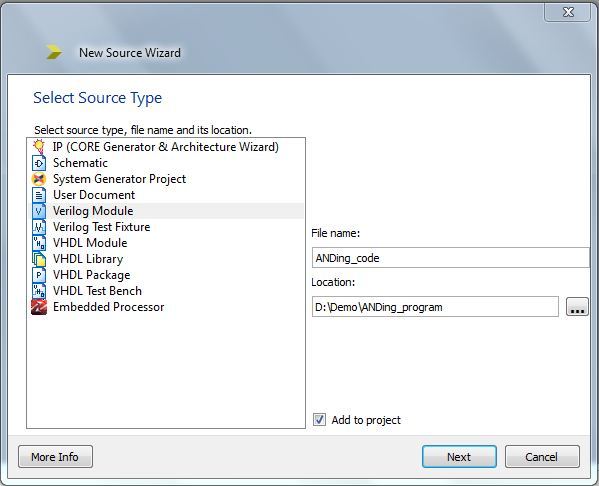


click on **Next**

**Project summary** window occurs.



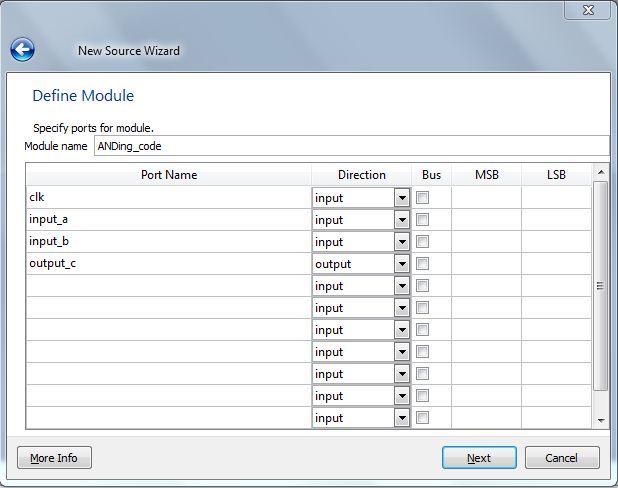
Select Source type is **Verilog Module** and enter the file name (**ANDing\_code**).

[](https://prashantbasargi.files.wordpress.com/2012/08/capture5.jpg)

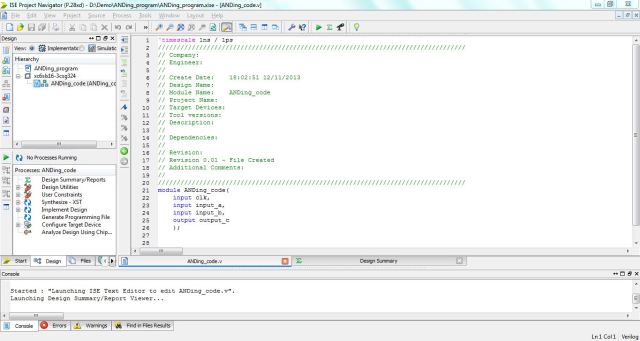
Click on **Next**

**3.3 DEFINE MODULE WINDOW**

Here we can define inputs & outputs and its bit/bus size. In ANDing example there are two inputs and output of single bit each. Also define clock signal for clocking operation.

[](https://prashantbasargi.files.wordpress.com/2012/08/capture61.jpg)

The Project Navigator window looks like below window.

[](https://prashantbasargi.files.wordpress.com/2012/08/capture82.jpg)

All the inputs and outputs are already defined in Define Module window so these inputs and outputs are seen in project navigator.

**3.4 Generating a Self-Checking Test Bench**

This process enables you to generate a self-checking HDL test bench equivalent to a test bench waveform (TBW) file and add the test bench to your project. You can also use this process to update an existing self-checking test bench.

The test bench generated by this process contains output data and self-checking code that can be used to compare the data from later simulation runs.

**Note**The [Add Test Bench to Project](https://www.xilinx.com/itp/xilinx10/isehelp/pp_p_process_add_test_bench_to_project.htm) process also adds a test bench to the project. However, the test bench produced by the Add Test Bench to Project process does not contain the extra self-checking test code contained in the test bench produced by the Generate Self-Checking Test Bench process.

You can use this process to automate the task of verifying simulation results. Rather than manually checking waveform results, you can generate a self-checking test bench that includes stimulus and predicted outputs. When running a simulation with this kind of test bench, simulation outputs will be monitored. If a difference is detected between the predicted and the actual outputs, an error is reported.

**Caution**The self-checking testbench produced by this procedure is only to be used for behavioral simulation. Using the self-checking testbench created for behavioral simulation to perform timing simulation can generate simulation errors.

1. Create a test bench waveform file.
2. Select **File > Save** to add the test bench waveform file to your project.
3. Edit your design and test bench waveforms as needed.
4. In the [Sources tab](https://www.xilinx.com/itp/xilinx10/isehelp/pn_r_sources_tab.htm), select **Behavioral Simulation** from the drop-down list.
5. Run behavioral simulation on your design with the test bench waveform file.
6. Repeat Steps 3 and 5 until you are satisfied with the simulation results. The design resulting from these steps is called the "golden" design.

**3.5 To Generate a Self-Checking Test Bench**

1. In the [Sources tab](https://www.xilinx.com/itp/xilinx10/isehelp/pn_r_sources_tab.htm), select the test bench waveform file for which you will generate the self-checking test bench.
2. In the [Processes tab](https://www.xilinx.com/itp/xilinx10/isehelp/pn_r_processes_tab.htm), expand **Xilinx ISE Simulator**, then expand**Simulate Behavioral Model.**
3. Right-click the **Generate Self-Checking Test Bench** process, and select **Properties**.
4. Set the property values in the Process Properties dialog box.
   * For the ISim, you can set the [ISim Properties](https://www.xilinx.com/itp/xilinx10/isehelp/pp_db_ise_simulator_properties.htm) in the Process Properties dialog box.
   * For the ModelSim simulator, you can set the [Simulation Properties](https://www.xilinx.com/itp/xilinx10/isehelp/pp_db_simulation_properties.htm) in the Process Properties dialog box.
5. Double-click **Generate Self-Checking Test Bench**.

A self-checking test bench is generated and added to your project.

The self-checking test bench is named *waveform\_file\_name*\_selfcheck\_beh.v (Verilog) or *waveform\_file\_name*\_selfcheck\_beh.vhd (VHDL). If you run the process again, you will be prompted to either overwrite the previous self-checking test bench or create a new file, iterating each time (*filename*\_0.vhd, *filename*\_1.vhd, etc.).

You can perform any of the following:

* Continue to refine your design and perform behavioral simulation using your original test bench waveform file.
* Perform behavioral simulation using your self-checking test bench any time design changes are made, to ensure that the changes do not affect the "golden" functionality.
* Synthesize the design.
* Write a program for in module present in project navigator.
* Right click on simulation sources and click on add sources. Now save the file with tb\_file name.
* Click on Project > New Source. Select Implementation Constraints file type and enter the file name.
* Write the inputs, outputs and its pin location in proper format of **.ucf** file.
* Double click on **Configure Target Device**and a new **ISE iMPACT** window open.

**Advantages of Verilog:**

• C-like syntax

• More compact code

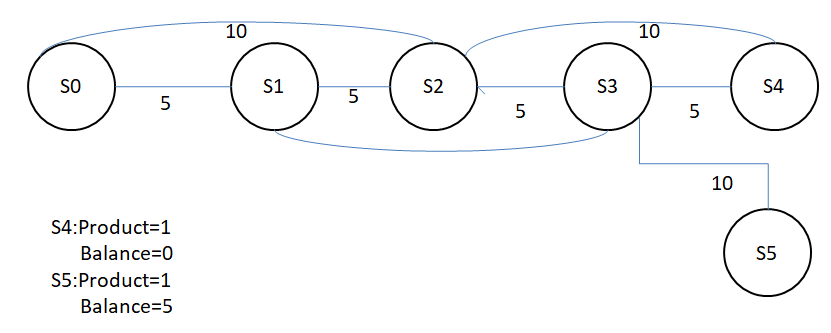
• Block commenting

• No heavy component instantiation as in VHDL

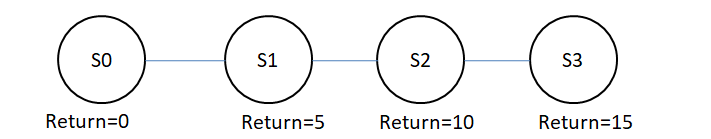
**CHAPTER 4**

**STATE DIAGRAMS**

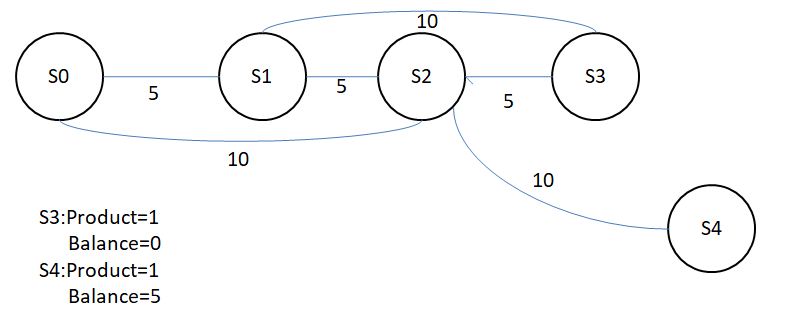
**4.1 FIRST YEAR**



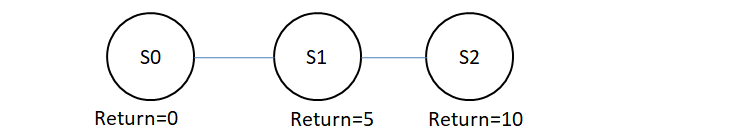
**4.1.1 CANCELLATION**



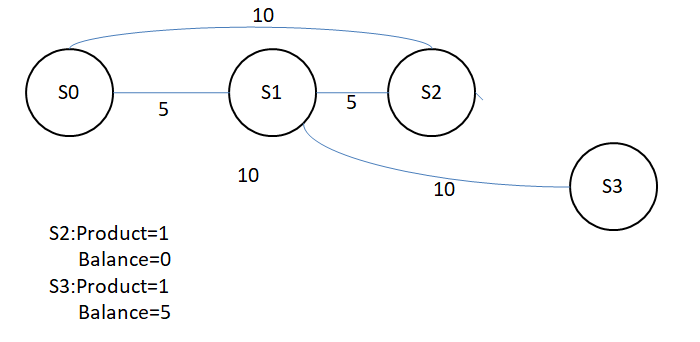
**4.2 SECOND YEAR**



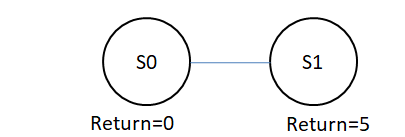
**4.2.2 CANCELLATION**



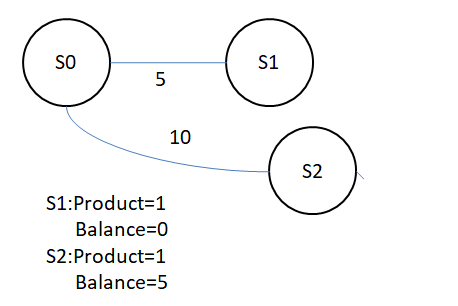
**4.3 THIRD YEAR**



**4.3.1 CANCELLATION**



**4.4 FOURTH YEAR**



**NOTE**: There is no cancellation for fourth year.

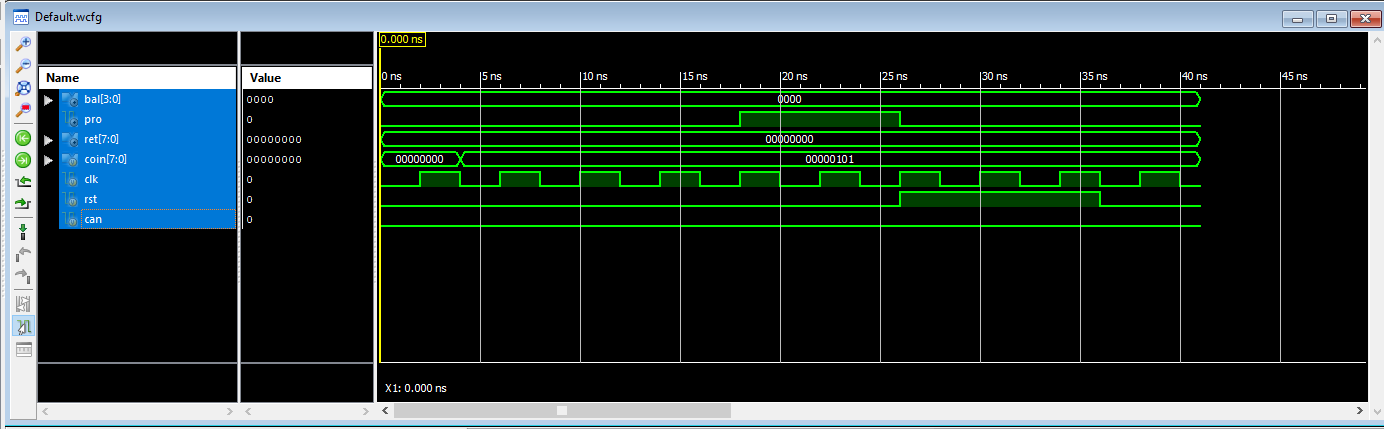
**CHAPTER 5**

**RESULTS**

**5.1 FIRST YEAR**

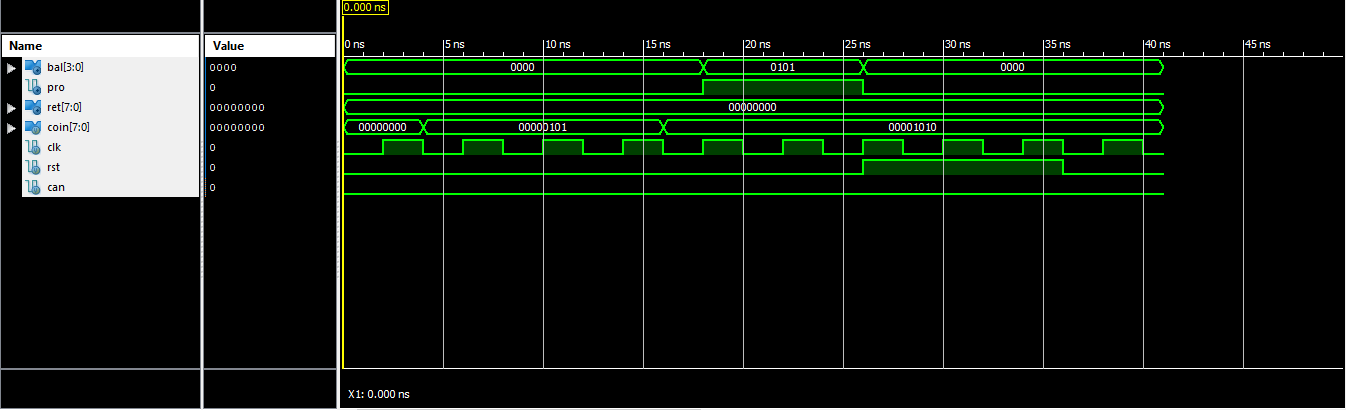
Inserted amount :Rs.20/-

Balance :Rs.0/-



Inserted amount :Rs.25/-

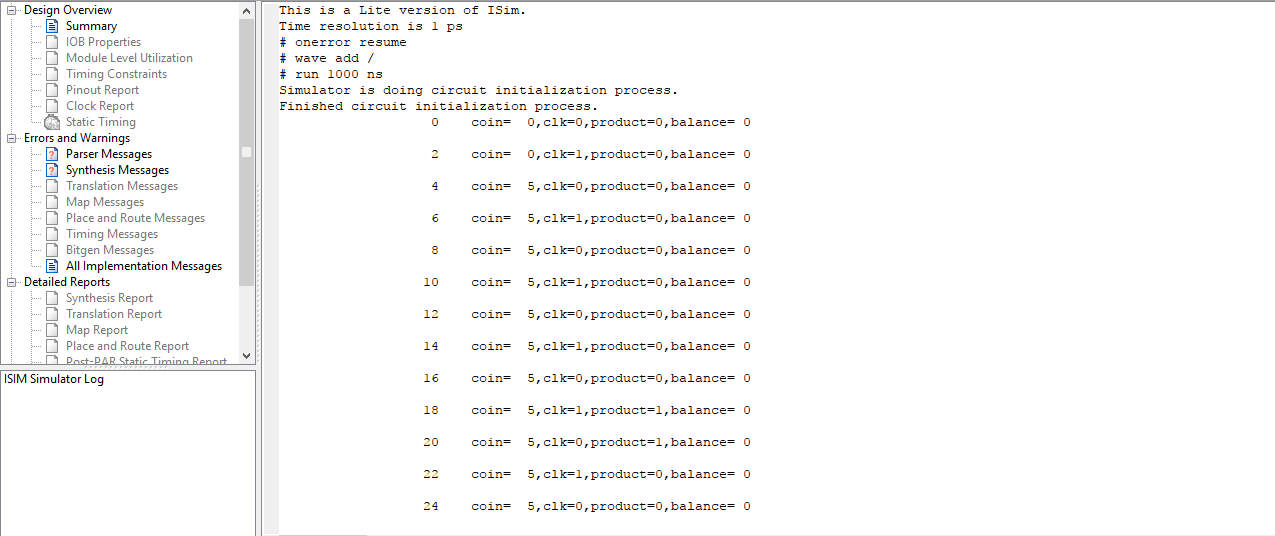
Balance :Rs.5/-



**5.1.1 TIMINGS**

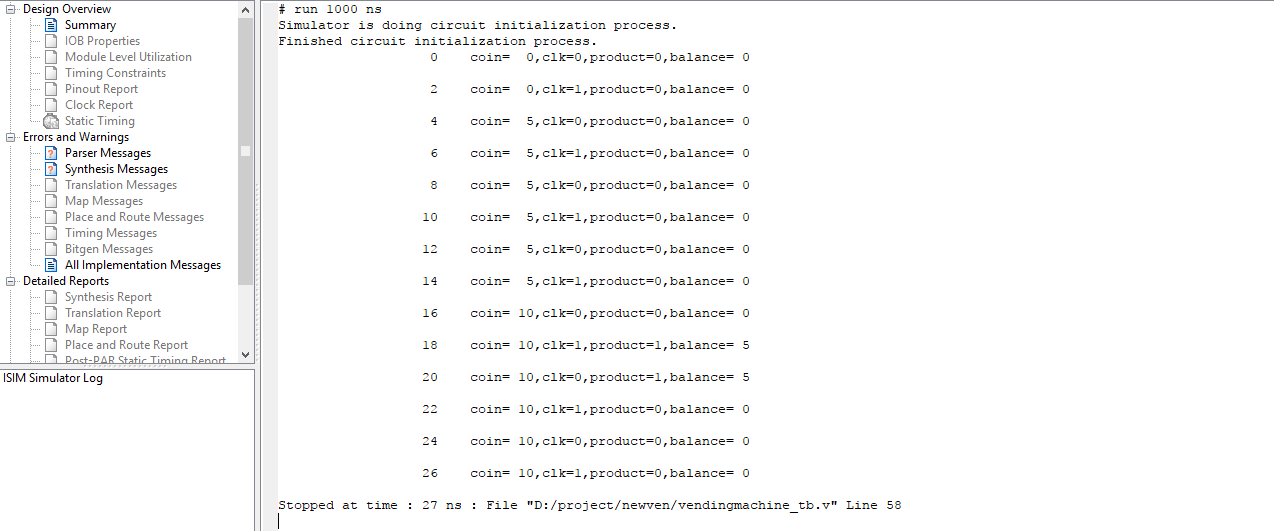
Inserted amount :Rs.20/-

Balance :Rs.0/-



Inserted amount :Rs.25/-

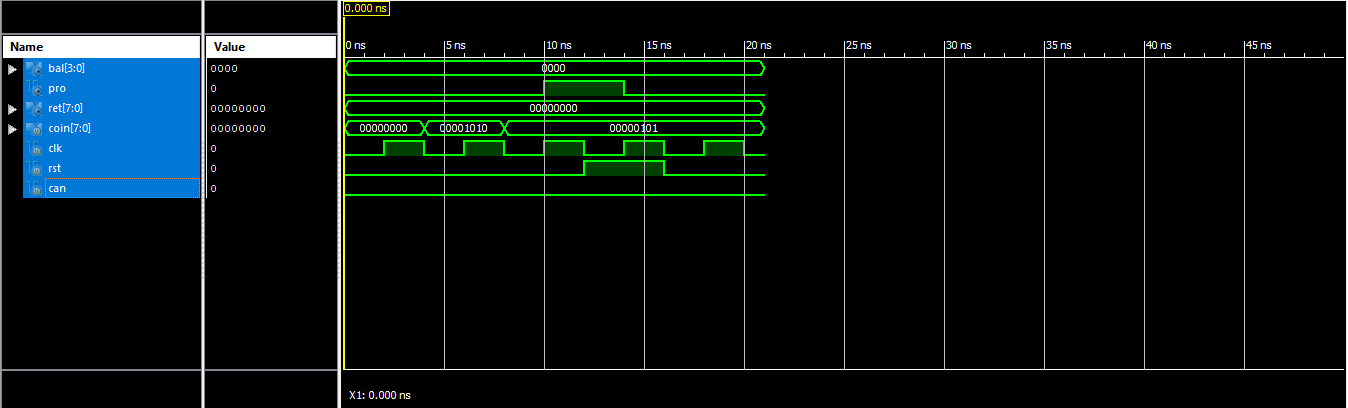
Balance :Rs.5/-



**5.2 SECOND YEAR**

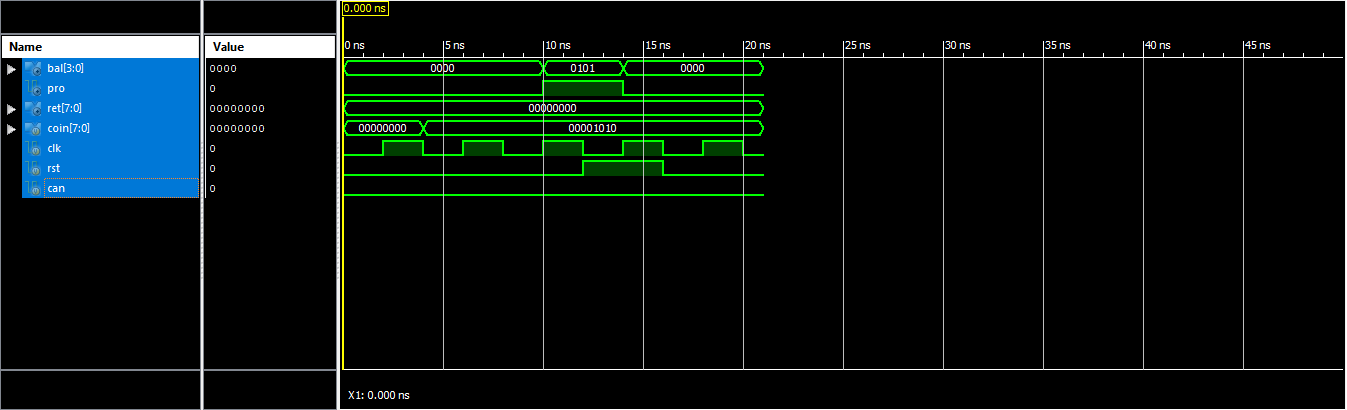
Inserted amount :Rs.15/-

Balance :Rs.0/-



Inserted amount :Rs.20/-

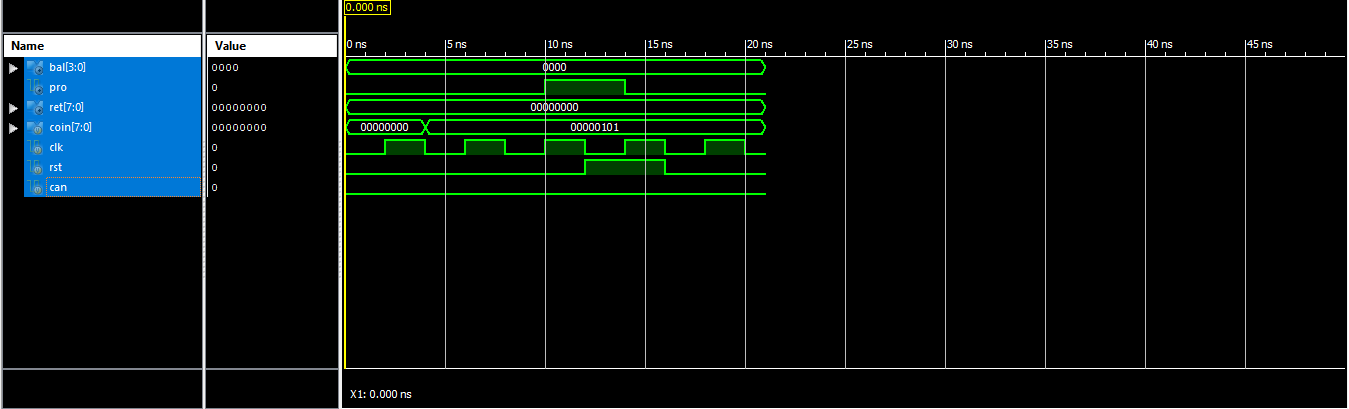
Balance :Rs.5/-



**5.3 THIRD YEAR**

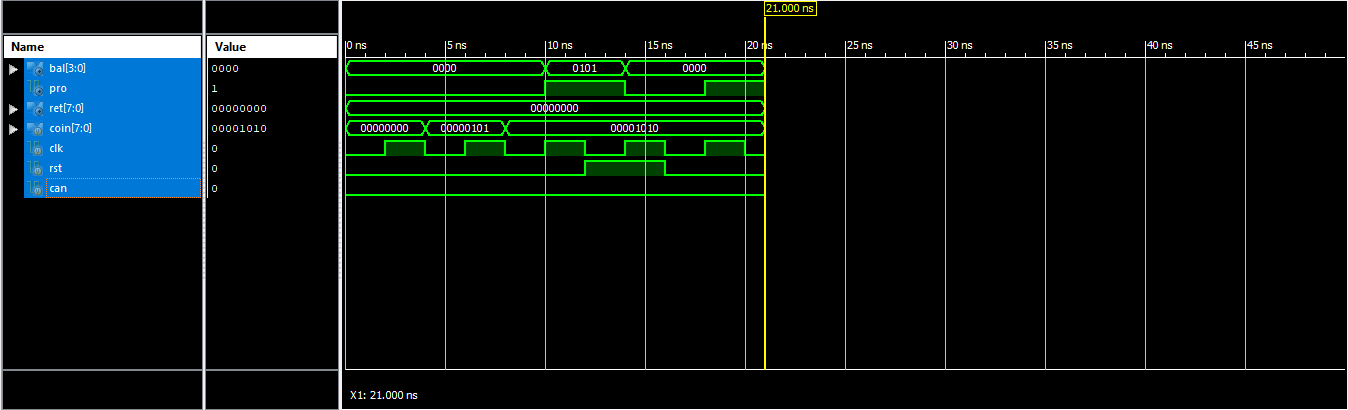
Inserted amount :Rs.10/-

Balance :Rs.0/-



Inserted amount :Rs.15/-.0

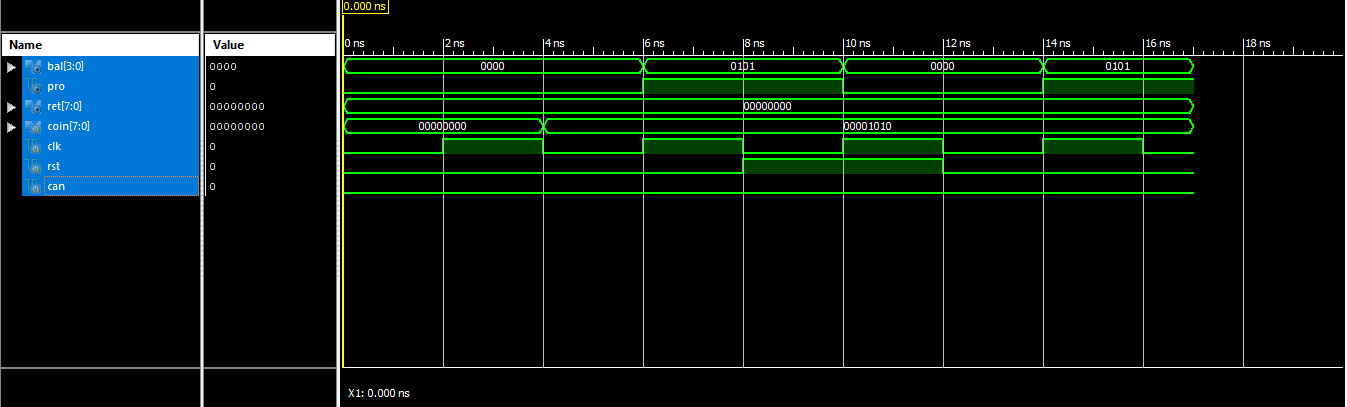
Balance :Rs.5/-



**5.4 FOURTH YEAR**

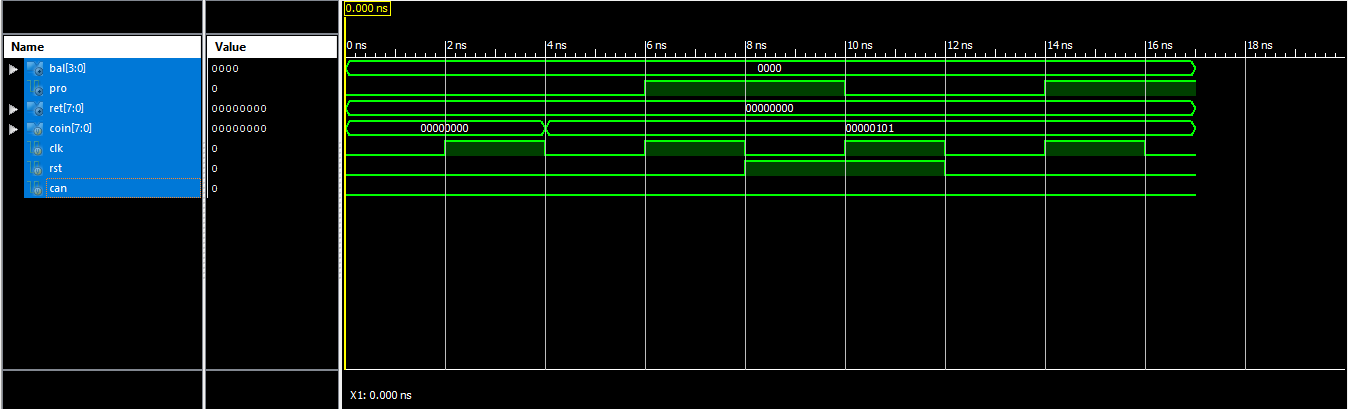
Inserted amount :Rs.5/-

Balance :Rs.0/-



Inserted amount :Rs.10/-

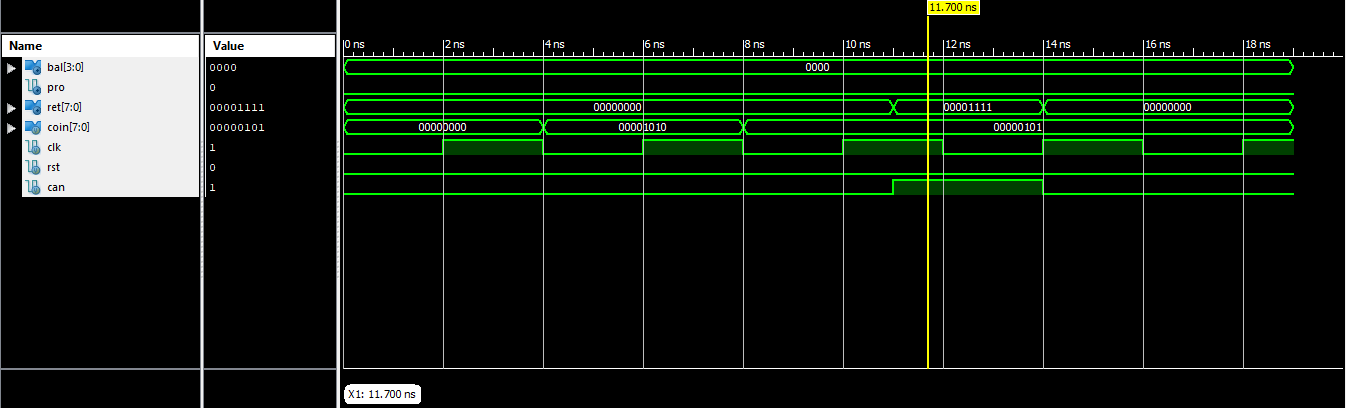
Balance :Rs.5/-



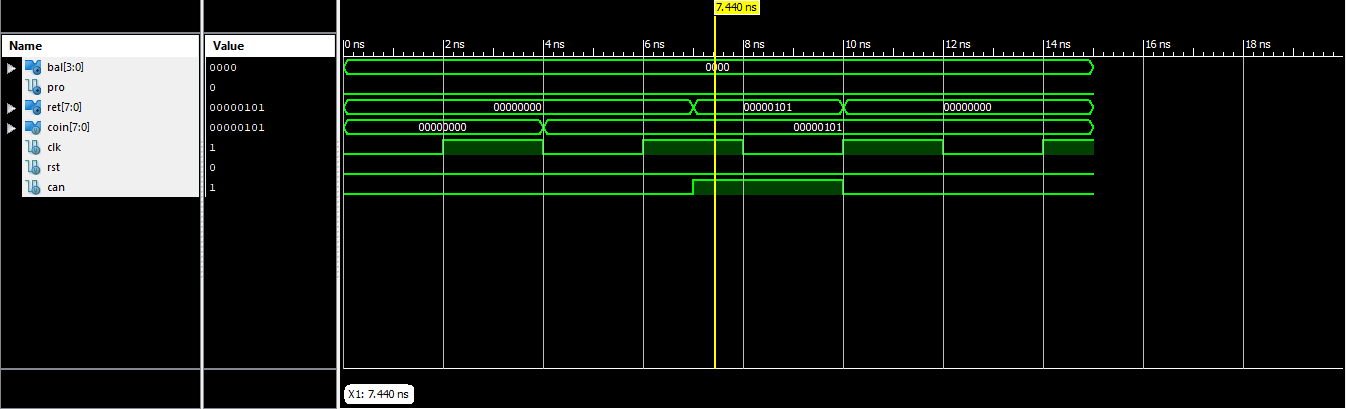
**5.5 CANCELLATION:**

**5.5.1 FIRST YEAR:**

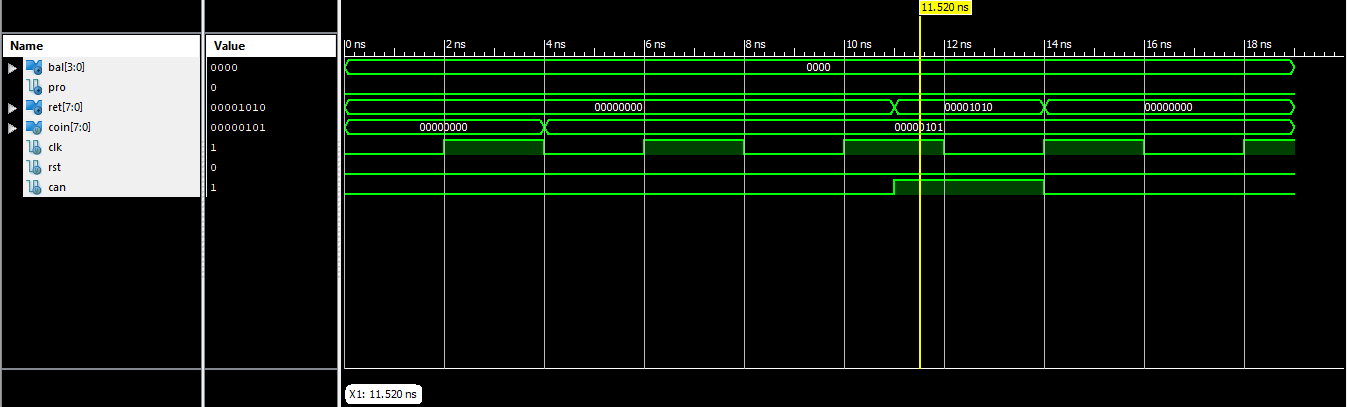
Return:Rs.15/-



Return:Rs.10/-

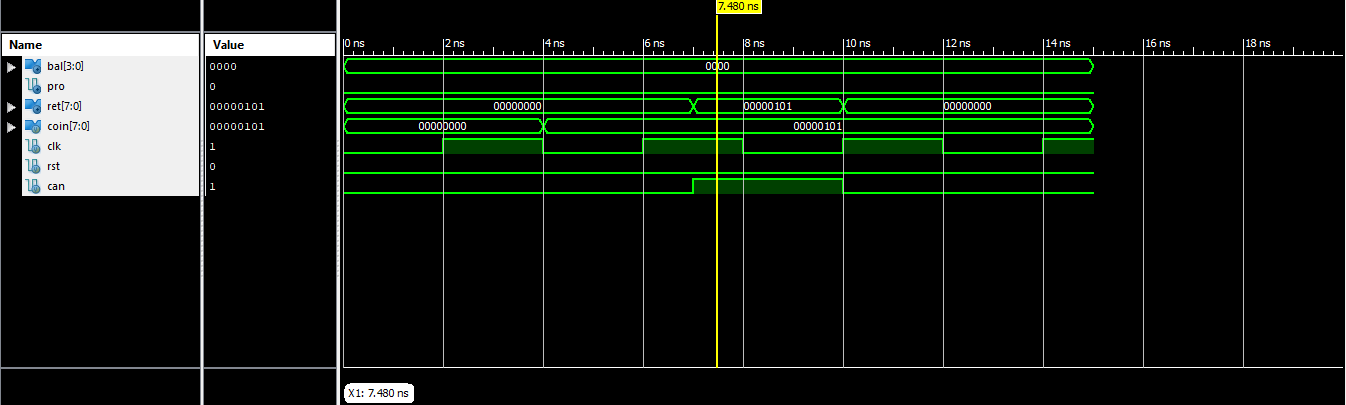


Return:Rs.5/-

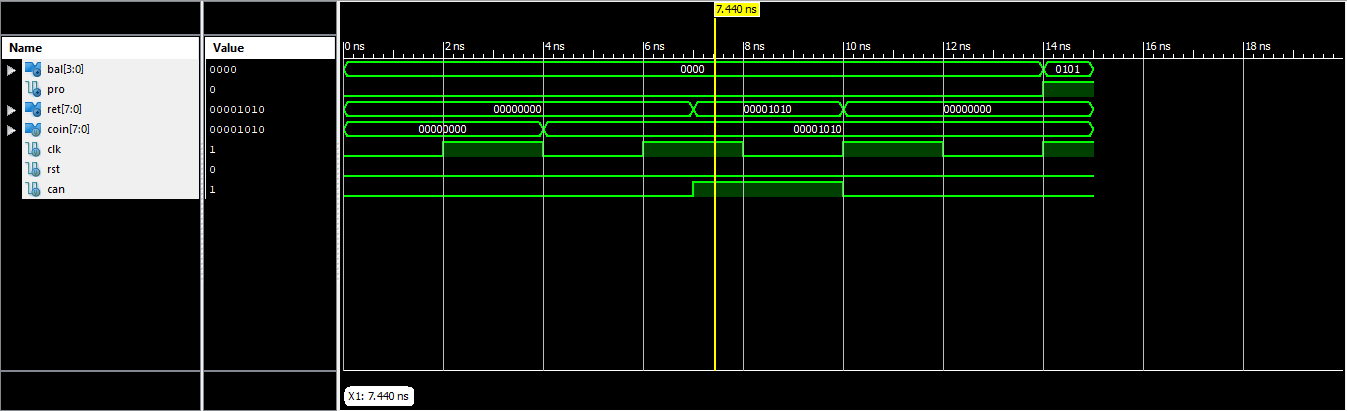


**5.5.2 SECOND YEAR:**

Return:Rs.10/-

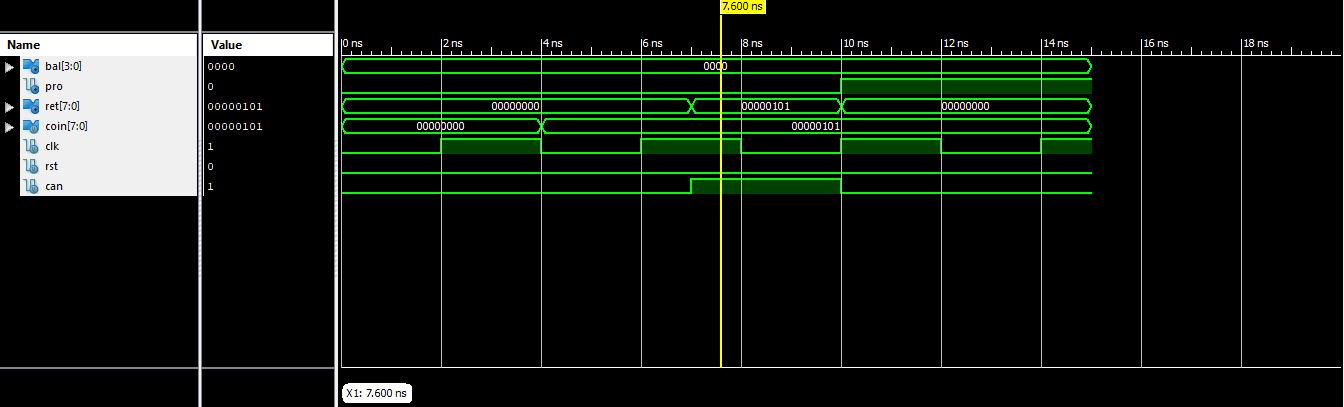
****

Return:Rs.5/-

****

**5.5.3 THIRD YEAR:**

Return:Rs.5/-



**5.5.4 FOURTH YEAR:**

As the cost price is equal to minimum amount, so there is no return of amount.

**CHAPTER 6**

**CONCLUSION**

As there will be Grouped Inputs present , the Subsidization process can performed.The process can be secured as there will be Selective Inputs. The process will be helpful to increase sales of product.It can be used in colleges where this model will be useful enough for students. It can also be used in various organizations such as Factories ,

Corporate Offices etc. Where subsidized rates can be given to experienced personnel with grouped inputs. The market of vending machines and consumables at growth level.In spite of having a tough competition, the demand of either consumables or vending machines is at increased level according to daily calls.

**CHAPTER 7**

**REFERENCES**

1. www.chicagotribune.com/dining/ct-vending-machine-for-homeless-20180103-story.html
2. Jaffe, Eric (Dec 2006). ["Old World, High Tech" (World's First Vending Machine, scroll down to read"](http://www.smithsonianmag.com/science-nature/old-world-high-tech-141284744/). Smithsonian.com. Retrieved 4 April 2017.
3. ["Automatic Vending Machines"](http://business.highbeam.com/industry-reports/equipment/automatic-vending-machines). Highbeam Business.
4. Kerry Segrave (12 July 2002), [Vending Machines: An American Social History](https://books.google.com/books?id=iCSqgzfy38QC), McFarland, [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [978-0-7864-8159-0](https://en.wikipedia.org/wiki/Special:BookSources/978-0-7864-8159-0)
5. ["What's "Tei-Park"?"](https://web.archive.org/web/20130629062547/http:/www.teipark.jp/e_about.html). Archived from [the original](http://www.teipark.jp/e_about.html) on 29 June 2013. Retrieved 5 June 2017.
6. Chandler, Alfred (2004). Scale and scope: the dynamics of industrial capitalism (1. paperback ed., 7. print. ed.). Cambridge, Mass.: Belknap Press. pp. 398–402. [ISBN](https://en.wikipedia.org/wiki/International_Standard_Book_Number) [978-0-674-78995-1](https://en.wikipedia.org/wiki/Special:BookSources/978-0-674-78995-1).
7. [National Automatic Merchandising Association](https://en.wikipedia.org/wiki/National_Automatic_Merchandising_Association). ["History of Vending and Coffee Services"](https://namanow.org/vending/history-of-vending-and-coffee-service). Retrieved 27 October 2016.
8. ["Cigarette vending machines banned in England"](http://www.bbc.co.uk/news/uk-15132529). [BBC](https://en.wikipedia.org/wiki/BBC). Retrieved 16 May 2014.
9. ["First Female Condom Machine Is Launched in France"](http://www.thebody.com/cdc/news_updates_archive/mar14_02/france_condoms.html). Retrieved 2007-03-25.
10. Holcombe, Madeline (21 April 2017). ["A vending machine at the UC Davis campus dispenses Plan B pills"](http://www.cnn.com/2017/04/21/health/davis-plan-b-machine-trnd/index.html). CNN.com. Retrieved 6 June 2017.