Innovation & Entrepreneurship Hub for Educated Rural Youth (SURE Trust – IERY)

VENDING MACHINE CONTROLLER

The domain of the Project:

RTL Design – Digital VLSI

Team Mentor :

Mr. Satish Devarapalli (Emulation Verification Engineer , Apple)

Team Members:

1. Ms.D.Susrutha --------- B.Tech, 4th year pursuing --- Team Leader
2. Ms.N.Keerthi --------- B.Tech, 4th year pursuing --- Team member
3. Mr. Ch.Jnaneswar --------- B.Tech 4th Year pursuing --- Team member
4. Mr. L.Nithin -------- M.Tech 2nd Year pursuing--- Team member

Period of the project

June 2025 to July 2025

Declaration

The project titled **"Vending Machine Controller"** has been mentored by **Satish sir,** organised by SURE Trust, from June 2025 to July 2025, for the benefit of the educated unemployed rural youth for gaining hands-on experience in working on industry relevant projects that would take them closer to the prospective employer. I declare that to the best of my knowledge the members of the team mentioned below, have worked on it successfully and enhanced their practical knowledge in the domain.

Team Members:	Signatures:
1.Ms.D.Susrutha	D. Susrutha.
2. Ms.N.Keerthi	N. Keerthi
3.Mr.Ch.Jnaneswar	Ch. Jnaneswar
4.Mr.L. Nithin	L. Nithin.

Mentor's Name : Mr. Satish Devarapalli
Designation— Emulation Verification Engineer , Apple

Prof. Radhakumari
Executive Director & Founder
SURE Trust

Table of contents

Executive Summary

This project involves the development of a **configurable, modular Vending Machine Controller (VMC)** using **Verilog HDL**. The system is designed to support up to **1024 distinct items**, each individually programmable for price and available quantity, making it highly scalable and adaptable to real-world vending scenarios. The VMC architecture operates in **two functional modes**:

1. **Configuration Mode** – Activated via an **Advanced Peripheral Bus (APB)** interface, this mode allows external agents to set or modify item parameters like cost and availability. Configuration is done dynamically without requiring a reset or interrupting the operational flow.
2. **Operational Mode** – In this mode, the system responds to **real-time user inputs** including item selection and currency insertion. It processes the transaction and performs dispensing if the conditions are met (i.e., item availability and sufficient funds).

   To support communication between the configuration logic (running on **pclk**) and the operational logic (running on **clk**), the system uses a **dual-port memory** with **clock domain crossing**. This ensures data consistency and safe synchronization across asynchronous clock domains. Core modules include:

- **Item Selection Logic**: Latches and validates user-selected item IDs.
- **Currency Accumulator**: Tracks total inserted currency and detects valid currency pulses.
- **Finite State Machine (FSM)**: Governs the control flow for selection, validation, currency handling, and dispensing.
- **Output Logic**: Handles item dispensing, availability decrement, and change calculation.

All operations are fully implemented in **hardware logic** with no reliance on software processors or external memory controllers. This results in **high-speed, deterministic behavior** suited for real-time applications. A comprehensive **Verilog testbench** has been developed to verify the design under different conditions including valid and invalid transactions, multiple item types, and reset behaviors. The testbench helps ensure **correctness, reliability, and functional coverage**.

This VMC design demonstrates a **robust, reusable, and efficient hardware solution** for embedded vending systems. Its **parameterized design** supports easy adaptation to different use-cases, making it suitable for **System-on-Chip (SoC)** platforms and commercial FPGA deployments.

Introduction

Background and context

Vending machines are common automated systems used to dispense products such as snacks, beverages, or tickets in exchange for currency. They operate without human supervision, making them ideal for high-traffic environments like offices, airports, railway stations, and educational institutions. Traditional vending machines use microcontrollers with fixed firmware, offering limited configurability and scalability.

As the demand for smarter and more adaptable vending systems grows, there is a need for hardware solutions that provide flexibility, real-time performance, and low power consumption. Hardware Description Languages (HDLs) like **Verilog** allow designers to implement such logic directly in hardware, particularly on platforms like **FPGAs or ASICs**, to achieve parallel processing and deterministic behavior.

This project introduces a **modular and configurable Vending Machine Controller (VMC)** built entirely in Verilog. The controller supports up to **1024 items** and features two operational modes: a **configuration mode** using the **AMBA APB interface**, and a **normal vending mode**. The configuration mode enables dynamic updates to item price and stock information, while the operational mode allows users to select items, insert currency, and receive change based on item cost.

To handle memory and synchronization across different clock domains (system and configuration), the architecture employs a **dual-port memory system**. The project integrates dedicated modules for **item selection, currency management, dispense logic, and control state machine**, ensuring a clean and efficient design. Real-time behavior is ensured by hardware-level decision-making without reliance on software or external processors.

The design has been verified using a Verilog **testbench**, simulating multiple operational scenarios to ensure correctness and robustness. This project demonstrates the application of **digital system design principles, FSMs, clock-domain crossing**, and **hardware interfacing**, offering a practical and scalable solution for vending applications.

Problem Statement

Modern vending machines often face challenges related to real-time responsiveness, configurability, and efficient resource utilization. Many commercial solutions rely on microcontroller-based software which may introduce delays, lack precise timing control, and make scalability difficult. These limitations affect user experience and system reliability, especially in embedded or FPGA-based deployments where deterministic performance is essential. There is a clear need for a hardware-centric vending machine system that offers modularity, fast operation, and configurable item management without the overhead of external processors or memory controllers. Additionally, a dual-mode system—allowing both real-time operation and easy configuration—is crucial for practical deployment and maintenance.

Key Functional Requirements:

1. Clock Domains: System clock: 100 MHz. , Configuration clock: 10 MHz.

2. Modes of Operation: Reset Mode: Resets all hardware elements.

Configuration Mode: Load item values, available counts, and reset dispensed counts using an APB interface.

Operation Mode: Accepts item selection and currency inputs, processes transactions, and handles item dispensing or change return.

3. Currency Input: Accepts multiple denominations: {5, 10, 15, 20, 50, 100}.Input is asynchronous to the system clock (10KHz to 50MHz).

4. Interfaces: APB-based configuration interface for setting up items.

Scope

The scope of this project is to design and implement a **modular, hardware-based vending machine controller** using **Verilog HDL**, suitable for deployment on FPGA or ASIC platforms. The system focuses on real-time performance, configurability, and hardware efficiency, eliminating the need for external processors or software control.

1. Technical Scope

Defines what the system is technically capable of doing.

- Implemented entirely in Verilog HDL targeting FPGA/ASIC platforms.
- Supports up to 1024 items using configurable memory addressing. Enables item selection, currency processing, and item dispensing through hardware logic.
- Includes a state machine controller to manage operational flow.
- Integrates an APB-based configuration interface for programming item price and count.

- Uses dual-port memory with clock domain crossing for safe data exchange between config and operation modes.
- Simulates real-world vending machine functionality using a Verilog testbench.

2. Functional Scope

Outlines what features and user-facing operations the system covers.

- **User Operation Mode**:
  Users can select items, insert currency, and receive items/change based on logic.
- **Configuration Mode**:
  Administrators can configure item data (price and count) using APB protocol.
- **Validation Logic**:
  Validates selection and ensures enough funds are inserted before dispensing.
- **Change Calculation**:
  Automatically computes and returns change to the user.
- **Inventory Management**:
  Tracks and updates item availability after each dispense.

3. Design Scope

Specifies design-related choices, constraints, and assumptions.

- Design assumes synchronous design practices, except for one clock domain crossing between APB and system logic.
- Does not include mechanical components (motors, sensors).
- Focuses strictly on digital logic control.
- Memory mapped structure and signals are synthesizable for hardware implementation.
- Design is limited to single item dispense per transaction.

4. Development Scope

Covers what was built or developed in this project.

- Verilog modules for:
  - `item_select`
  - `currency_val`
  - `main_controller` (FSM)
  - `config_block` (APB Interface)
  - `item_memory`
  - `output_logic`
- Fully working testbench for simulation and verification.

Limitations

1. Hardware Implementation Limitations

The design has been verified only through simulation and has not been deployed or tested on an actual FPGA or ASIC platform.Real-world physical constraints like signal integrity, I/O voltage levels, and thermal considerations are not addressed.

2. Currency Handling Constraints

The system supports a single fixed currency format with no dynamic handling of multiple denominations or international currencies.No provision exists for accepting coins or notes with physical validation mechanisms.

3. Fixed Currency Denominations

Only specific denominations are supported: 5, 10, 15, 20, 50, and 100 INR.The design cannot handle coins or custom currency values without modifying the RTL.

4. Single Item Dispense per Transaction

The controller supports dispensing only one item per transaction.Multiple item purchases in a single session are not supported, requiring repeated item selections.

5. Limited User Interaction Handling

No support for:Transaction cancellation after currency insertion.Timeouts if a user stops inserting currency midway.Authentication or payment via digital methods (e.g., cards, QR codes).

6. No Security or Fault Detection Logic:

The design does not include:Tamper detection, Fake note detection Error handling for failed dispense mechanisms

7. Asynchronous Input Handling Complexity:

Currency and item selection inputs are asynchronous to the system clock, which may require synchronizer logic that can be sensitive to metastability if not carefully designed.

8. No Real-Time Monitoring or Logging:

No provision for real-time transaction logging or external monitoring/debug interfaces. Tracking is limited to internal counters for dispensed and available items.

Innovation

1. Real-Time FSM Integration

The project goes beyond basic finite state machine (FSM) implementation by supporting dynamic transitions based on live inputs. This ensures timely responses to user actions, mimicking real-world vending logic. The FSM is responsible for tracking state progression from selection to dispense.

2. Dynamic Coin Handling

The system allows insertion of multiple coin denominations in any order. It adds each value to a running total without requiring exact change upfront. This flexibility improves usability and user convenience.

3. Smart Change Calculation

If the inserted amount exceeds the item price, the system automatically calculates and returns the balance. This mirrors real vending machines, improving accuracy and realism. It enhances the user experience by preventing currency loss.

4. Transaction Cancel Feature

A cancel function enables the user to abort the transaction at any stage. Once canceled, the system clears selection and returns the inserted coins. This feature adds reliability, trust, and control to the user interaction.

5. Modular Hardware Design

The architecture is composed of clearly defined modules such as input logic, memory, controller, and output logic. Each module is reusable, testable, and independently verifiable. This modularity simplifies debugging, enhances scalability, and supports future upgrades.

Project objectives

1. Develop a Configurable Vending Machine Controller IP

Objective:
Design and build a flexible, parameterized Verilog-based controller IP to manage core vending operations such as item selection, currency input, and dispensing. The controller must support scalability and modularity for integration into various vending platforms.

Expected Outcome:
A reusable and synthesizable IP core written in Verilog HDL capable of handling up to 1024 different items and supporting standard bus interfaces such as APB for configuration.

2. Implement Dual-Mode Operation (Configuration & Vending)

Objective:
Enable the controller to operate in two distinct modes: a configuration mode for updating item data (price, availability) and a normal mode for executing user transactions.

Expected Outcome:
A seamless transition mechanism between modes using FSM logic, ensuring that item memory updates can be done securely without affecting real-time operation.

3. Design Robust Currency Handling and Validation Logic

Objective:
Create a module that accumulates currency values from user inputs, verifies their validity, and determines whether the total meets or exceeds the item cost.

Expected Outcome:
A validated currency system that supports partial and full payment handling, correctly manages overpayment by calculating change, and blocks invalid transactions.

4. Implement Smart Dispense Control with FSM Logic

Objective:
Introduce a finite state machine to control the dispense logic, ensuring transactions follow correct state transitions such as selection, validation, dispense, and reset.

Expected Outcome:
A predictable and state-driven vending controller that improves user experience by controlling timing and logic flow, enabling a fault-tolerant transaction cycle.

5. Design a Parameterized Dual-Port Memory

Objective:
Develop internal memory capable of holding item prices and available counts, accessible simultaneously by both configuration and operation logic across clock domains.

Expected Outcome:
A dual-port memory block that safely supports APB write access and operational read access, maintaining item integrity and preventing race conditions.

6. Support Clock Domain Crossing for Safe Operation

Objective:
Allow configuration through the APB interface (running on `pclk`) while maintaining system operation on a separate `clk`, ensuring data coherency between domains.

Expected Outcome:
A design that cleanly separates clock domains using synchronized memory access, latches, and handshaking mechanisms, verified through simulation.

7. Enable Full Testbench Verification and Debug Support

Objective:
Develop a testbench that simulates real-world scenarios, including valid/invalid selections, underpayment, overpayment, and no inventory conditions.

Expected Outcome:
A functional testbench with clear `$display` outputs showing all internal signals and logic states, validating the design before synthesis or implementation.

8. Deliver a Resource-Efficient, Synthesizable Design

Objective:
Ensure that the design consumes minimal resources in terms of logic gates, registers, and memory, making it suitable for deployment on low-cost FPGAs or embedded SoCs.

Expected Outcome:

A compact RTL design that passes synthesis checks, consumes limited area, and achieves low power, making it ideal for portable and embedded vending applications.

Methodology and Results

Methods/Technology Used

1. RTL Design Using Verilog HDL

The core of the vending machine controller was designed using Verilog, a hardware description language suited for synthesizable digital systems. Modular and parameterized design allowed the system to be reusable, readable, and scalable up to 1024 items.

2. Modular Architecture

The system was broken down into functional blocks—such as item selection, currency handler, output logic, and configuration interface. Each module handled a specific task, enabling easier debugging and independent simulation.

3. Finite State Machine (FSM) Control Logic

The vending machine's behavior was governed by FSMs, which controlled sequencing of item validation, payment handling, and item dispensing. This enabled deterministic and predictable system behavior under all input conditions.

4. APB-Based Configuration Interface

An APB (Advanced Peripheral Bus) protocol was implemented to allow configuration of item prices and stock during run-time. This enabled easy integration with processors or microcontrollers in embedded systems.

5. Dual-Port Memory Design

Item information such as price, availability, and dispense ID was stored in a synchronous dual-port memory. This allowed safe access to memory from two different clock domains (system and APB) with proper synchronization.

6. Clock Domain Crossing (CDC)

Since the system operated with two separate clocks (clk and pclk), careful handling of signals across clock domains was ensured. Register synchronization was used to avoid metastability and ensure reliable data transfer.

7. Parameterization for Scalability

Parameters such as `ITEM_ADDR_WIDTH`, `CURRENCY_WIDTH`, and `MAX_ITEMS` were defined to allow the system to be scaled for different product ranges and currency formats, supporting flexibility in deployment.

8. Simulation-Based Verification

A dedicated testbench was developed to simulate various user operations including item selection, currency input, and cancellation. Waveform analysis verified correctness of the FSM transitions, output logic, and change handling.

Tools/Software Used

1. Xilinx Vivado Design Suite-Used for RTL design, synthesis, implementation, simulation, and verification of the vending machine controller. Vivado provided an integrated environment for constraint management, timing analysis, and bitstream generation.

2.EDA Playground (for Remote Simulation)

A cloud-based platform used for quick simulations and sharing of Verilog code. It supports multiple simulators like Icarus Verilog, providing portability in collaborative environments.

Project Architecture

Vending machine controller is a digital IP core utilized in managing item selection, money handling, and item dispensing functions of a vending machine. Variable item setup and setup and monitoring are facilitated through the design, and control is provided via an AMBA APB interface.

1.System Clock & Reset Initialization

The system begins with a 100 MHz system clock for real-time vending and a
10 MHz APB clock for configuration. The active-low reset (prstn) is utilized to reset
all the internal registers, memories, and FSMs.

2.Architectural Modes

The controller operates in three main modes:

Mode	Condition	Description
Reset Mode	rstn = 0	The rstn pin is low and other pins are don't care
Config Mode	rstn = 1,cfg_mode = 1	Host (vendor) loads item values and stock through the APB interface
Operation Mode	rstn = 1,cfg_mode = 0	Handles currency input, item selection, and vending logic

3. Configuration Mode via APB Interface

In **configuration mode** (`cfg_mode = 1`), an external processor or vendor host configures the vending machine via a standard **AMBA APB (Advanced Peripheral Bus)** interface. The configuration includes:

- Setting individual item costs.
- Initializing available item stock.
- Resetting internal dispense counters.

All configuration commands are clocked by `pclk` and utilize standard APB signals: `psel`, `pwrite`, `paddr`, `pwdata`, and `prdata`. The APB logic drives the `mem_we`, `mem_waddr`, and `mem_wdata` signals, which update the internal item memory.

4. Dual-Port RAM for Item Configuration

A **parameterized dual-port memory** (`item_memory`) holds the configuration of all items. Each 32-bit memory entry stores:

- **Bits [15:0]**: `item_price` – Cost of the item in rupees.
- **Bits [23:16]**: `avail_count` – Available stock count.
- **Bits [31:24]**: `dispensed_item` – Total dispensed count.

The memory has two access ports:

- **Port A** (APB Clock): Used during configuration for writing and reading values.
- **Port B** (System Clock): Used during runtime for accessing selected item info.

This dual-port design ensures **synchronized access across clock domains** and enables configuration and operation to proceed independently.

5. Asynchronous Item Selection Interface

Users interact with the system by selecting items using the `item_select` and `item_select_valid` signals. Since user inputs can originate from slower interfaces (e.g., physical buttons, microcontrollers), these signals are **synchronized to the 100 MHz clock domain** before processing. A validation FSM ensures the selected item ID is within the configured range and checks its availability.

6. Currency Input and Accumulation

The vending machine supports multiple currency denominations (e.g., ₹5, ₹10, ₹20, ₹50). Users input currency using `currency_value` and `currency_valid` signals, which are similarly asynchronous. A dedicated currency accumulation block:

- Synchronizes the input to the system clock.
- Adds each input to the `total_currency` register.

- Monitors when the total reaches or exceeds the selected item cost.

7. Vending Decision Logic (FSM in Operation Mode)

When in **operation mode** (`cfg_mode = 0`) and valid input is detected, a finite state machine (FSM) controls vending logic:

- Checks if total inserted currency ≥ item cost.
- Verifies that `avail_count > 0`.
- If valid: Dispenses item, calculates and returns change, and updates counters.
- If invalid: Returns inserted currency and signals an error.

This FSM ensures transactional accuracy, safety, and responsiveness under all input scenarios.

8. Output Signal Generation

Upon successful vending:

- `item_dispense_valid` is asserted for **one system clock cycle** to notify the external environment.
- `item_dispense` holds the ID of the dispensed item.
- `currency_change` reflects the returned change value.
  These outputs can be sampled by external systems for logging or feedback mechanisms.

9. Inventory Tracking and Status Calculation

After each successful transaction, the internal memory is updated to reflect:

- `dispensed_item` is incremented.
- `avail_count` is decremented.

A configuration register maintains the number of configured items and supports live stock monitoring. The APB interface can read these values to display or log real-time stock status.

Transaction Flow Summary

Configuration Flow (Vendor Controlled)

1. Set `cfg_mode = 1`.
2. Use APB to:
   - Set total number of items.
   - For each item: set cost, availability, and reset dispense count.

User Flow (Runtime)

1. User selects item using `item_select`.
2. Inserts currency using `currency_value`.
3. Currency is accumulated until it's sufficient.
4. FSM validates transaction:
   - If valid → item dispensed, change returned.
   - If invalid → currency returned, error signaled.

Module Responsibilities

Module Name	Functionality
item_select	Captures the user's item selection, validates the input, and generates `item_selected` and `selection_valid` signals.
currency_val	Accumulates and synchronizes asynchronous currency input, computes total value, and asserts `currency_avail` when sufficient currency is inserted.
main_controller	Acts as the central FSM managing mode transitions and vending logic, enabling dispensing through `dispense_enable`.
output_logic	Generates final vending outputs like `dispense_valid`, `item_dispensed`, and `currency_change` based on internal logic and FSM decisions.
config_block	Interfaces with the host processor via the APB protocol for configuring item memory (`mem_we`, `mem_waddr`, `mem_wdata`).
item_memory	Dual-port memory storing item price, stock, and dispense count; accessible by both APB and vending logic.
vending_machine_top	Integrates all modules into a top-level RTL IP, handling signal routing, synchronization, and complete vending machine functionality.

Registers used:

I. **Item_select:** It is a register which is used to store the value of the item which we want to select. It is further used in the configuration mode and the operation mode.

II. **selection_valid:** It is a register flag which is used to check if the item we selected is a valid one or not. It indicates logic "1" for one clock cycle when the item selected is a valid one and indicates logic "0" when the item selected is invalid.

III. **total_currency:** It is register which is used to store the total value of sum of all the currencies inserted so far. which is then compared with the item cost when the selected item is available.

IV. **currency_avail:** It is a register which is used to indicate that a valid currency is inserted. It indicates logic '1' for one clock cycle when a valid currency is inserted and indicates logic '0' when a invalid currency or no currency is inserted.

V. **pready:** It is a one bit register used to write and read the data values at the selected address. The signal become active high for one clock cycle for each new value of data while doing write and read operations.

VI. **mem_we:** When the register mem_we is active high it undergoes write operation or else it is in read operation.

VII. **mem_waddr:** Stores the address of the selected item while writing.

VIII. **mem_wdata:** Stores the data that is written while doing write operation.

IX. **mem_raddr:** Stores the address from where the data has to be read in read operation.

X. **item_price:** It is a 16-bit register used to store the value of the item selected.

XI. **avail_count:** It is a 8-bit register which stores the total available count when an item is selected.

XII. **dispense_valid:** It is used to denote that the item selected is ready to dispense. It is active high for one clock cycle when the item is ready to dispense.

XIII. **item_dispensed:** It is used to store the value of the item that has to be dispensed. Which then used to dispense that item at the output.

XIV. **currency_change:** It is a register which stores the value of the change after deducting the item cost from the total amount inserted and then the change is dispensed after the item dispensation.

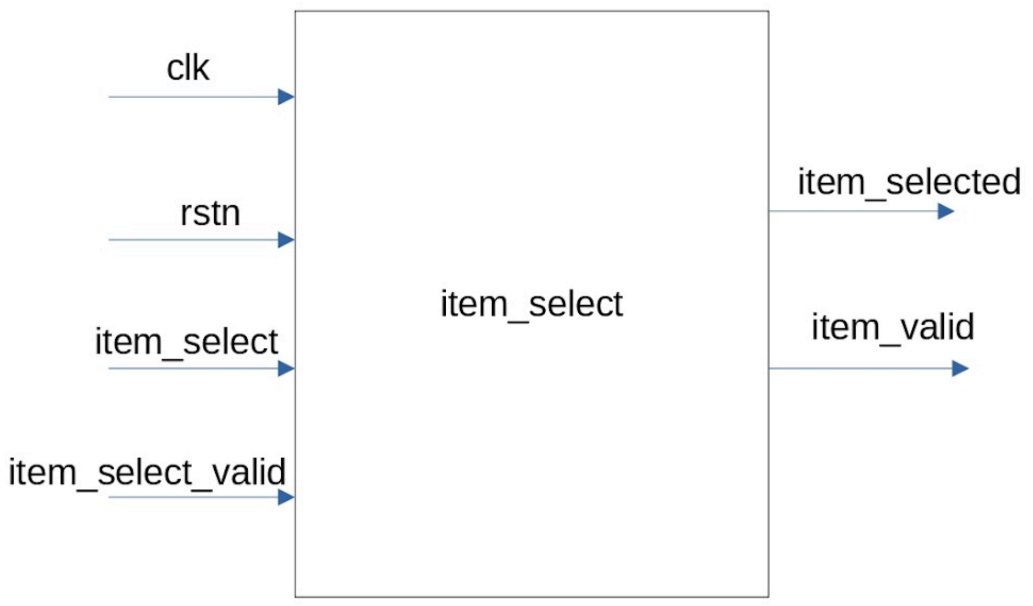
XV. **cfg_mode_ff1, cfg_mode_ff2:**

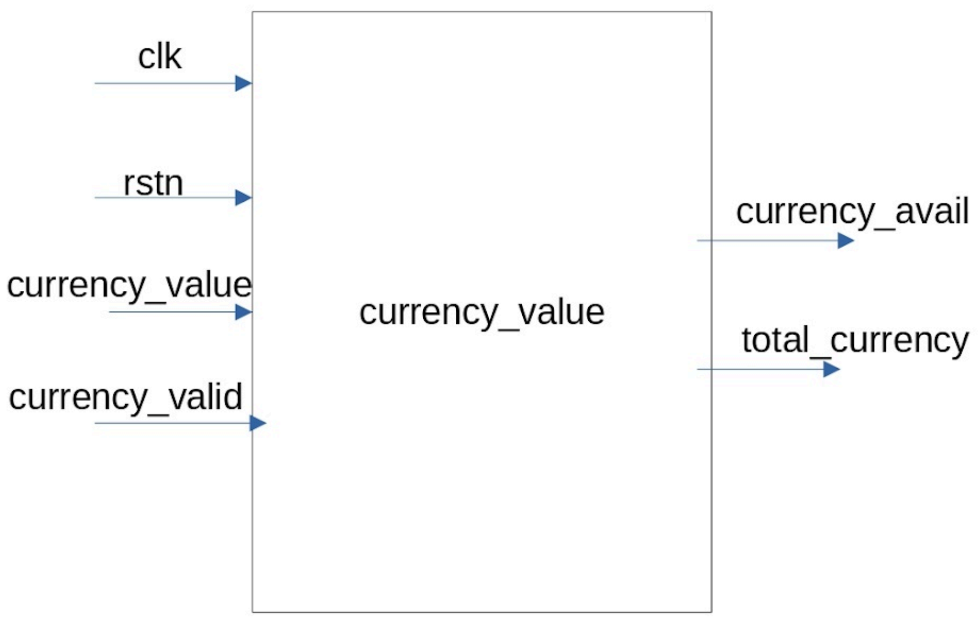
They are 1-bit data flipflops used to synchronize the clock in configuration block to overcome Clock Domain Crossing while writing and reading data.

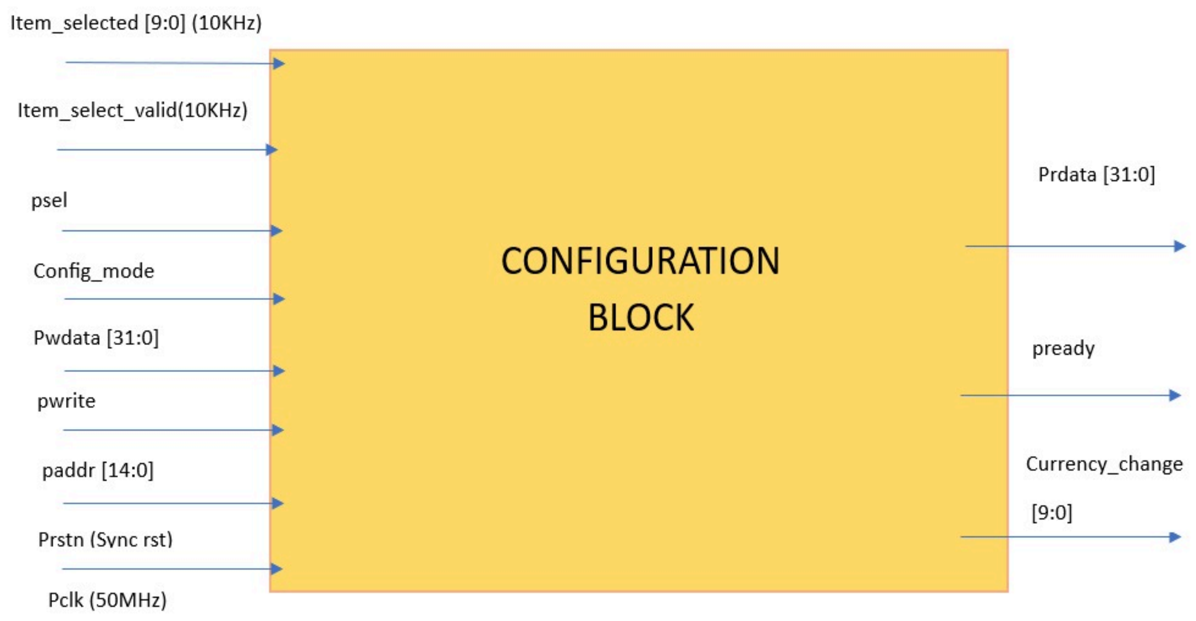
currency_valid_sync_0, currency_valid_sync_1:

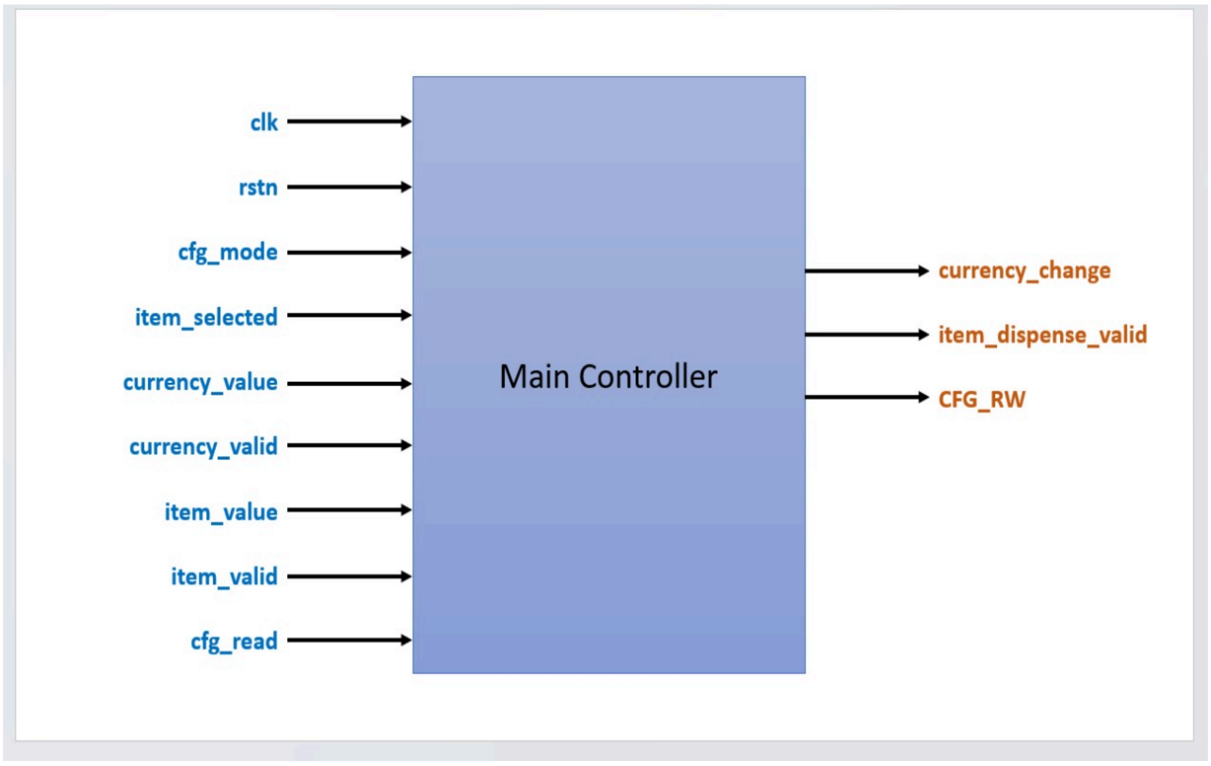
They are 1-bit data flipflops used to synchronize the currency insertion and currency valid checking.

Input_block

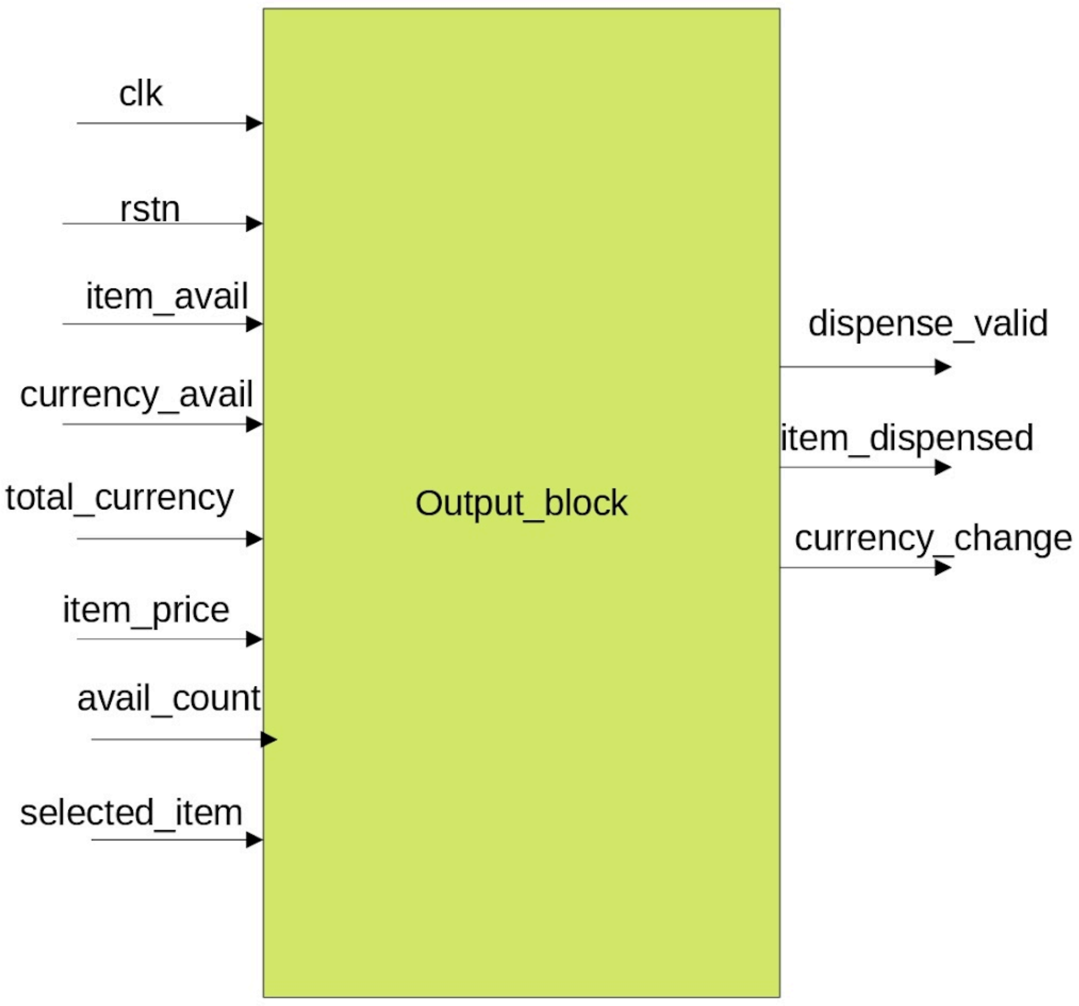

clk
rstn
item_select
item_select_valid
item_select
item_selected
item_valid


clk
rstn
currency_value
currency_valid
currency_value
currency_avail
total_currency


Item_selected [9:0] (10KHz)
Item_select_valid(10KHz)
psel
Config_mode
Pwdata [31:0]
pwrite
paddr [14:0]
Prstn (Sync rst)
Pclk (50MHz)
CONFIGURATION BLOCK
Prdata [31:0]
pready
Currency_change [9:0]

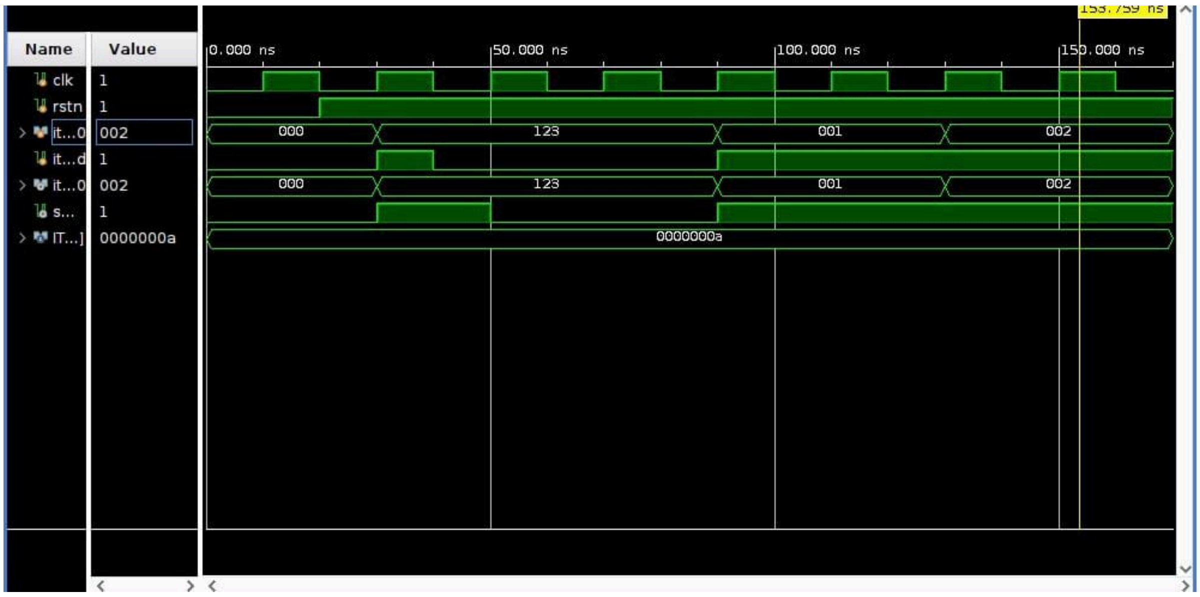

clk
rstn
cfg_mode
item_selected
currency_value
currency_valid
item_value
item_valid
cfg_read
Main Controller
currency_change
item_dispense_valid
CFG_RW

Output_block

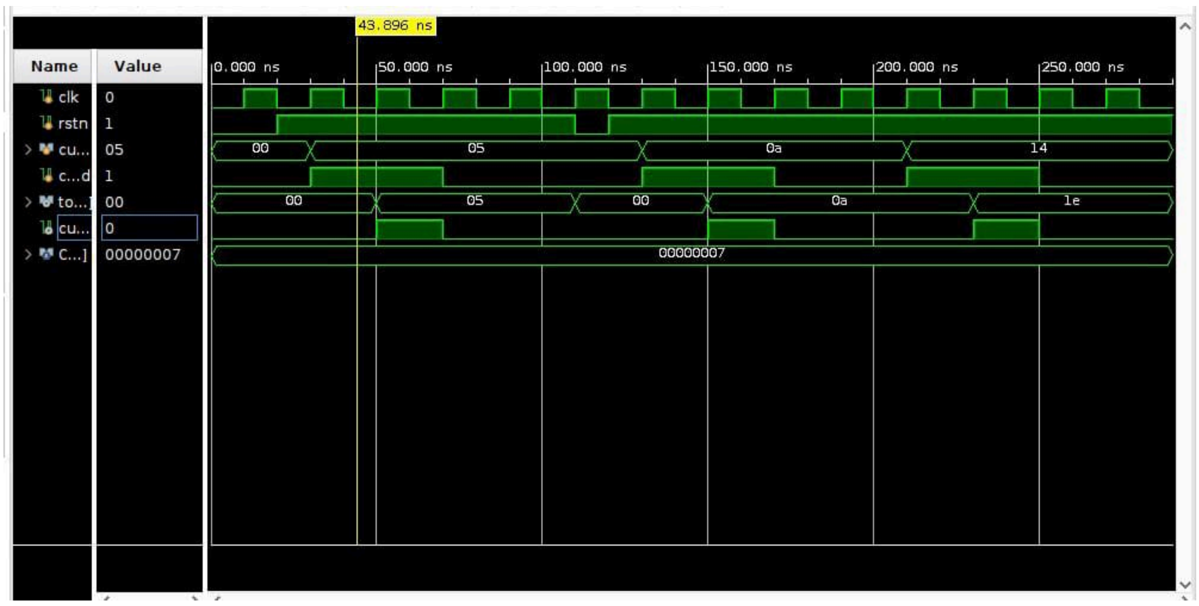


Results

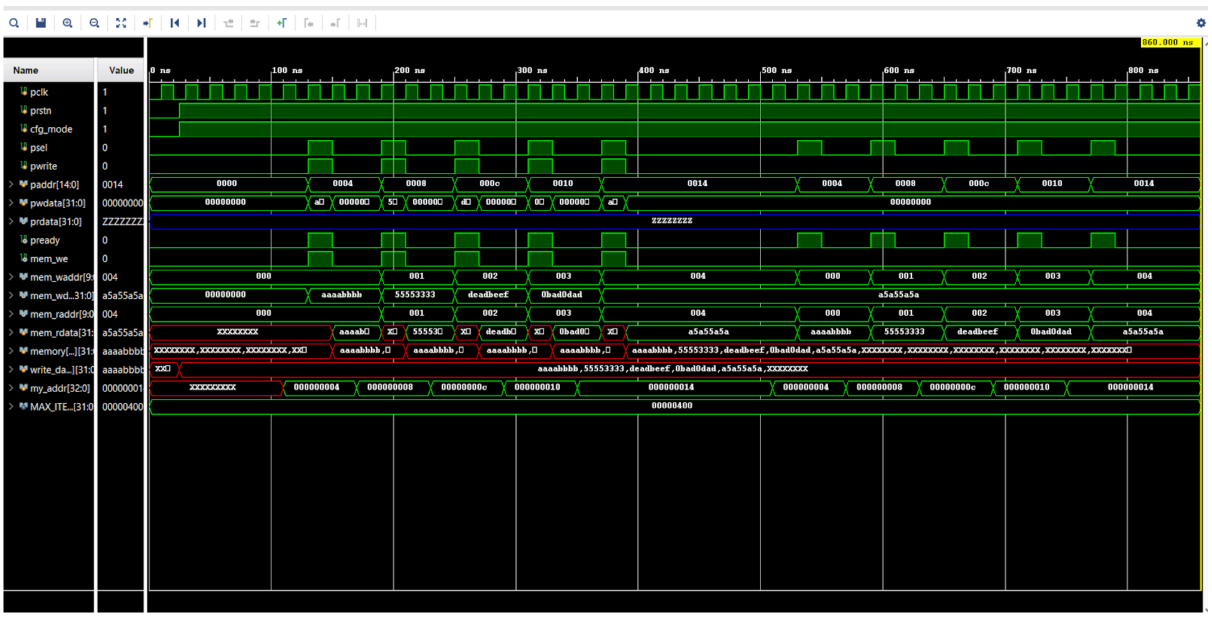
1.Item_select



2.Currency_val

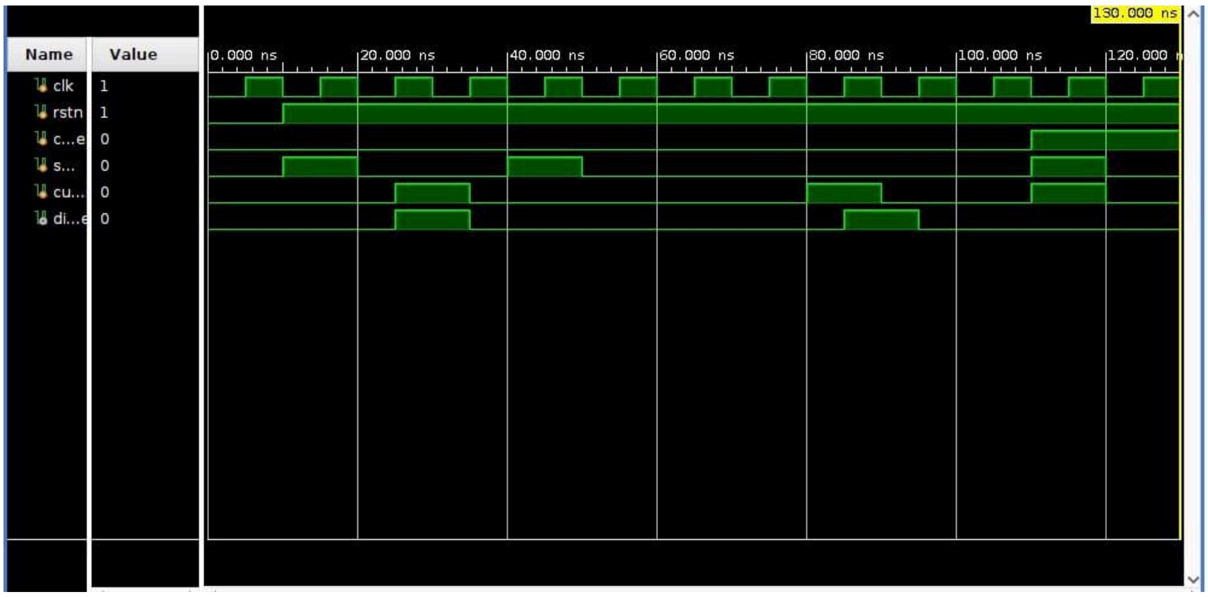


3.configure_block

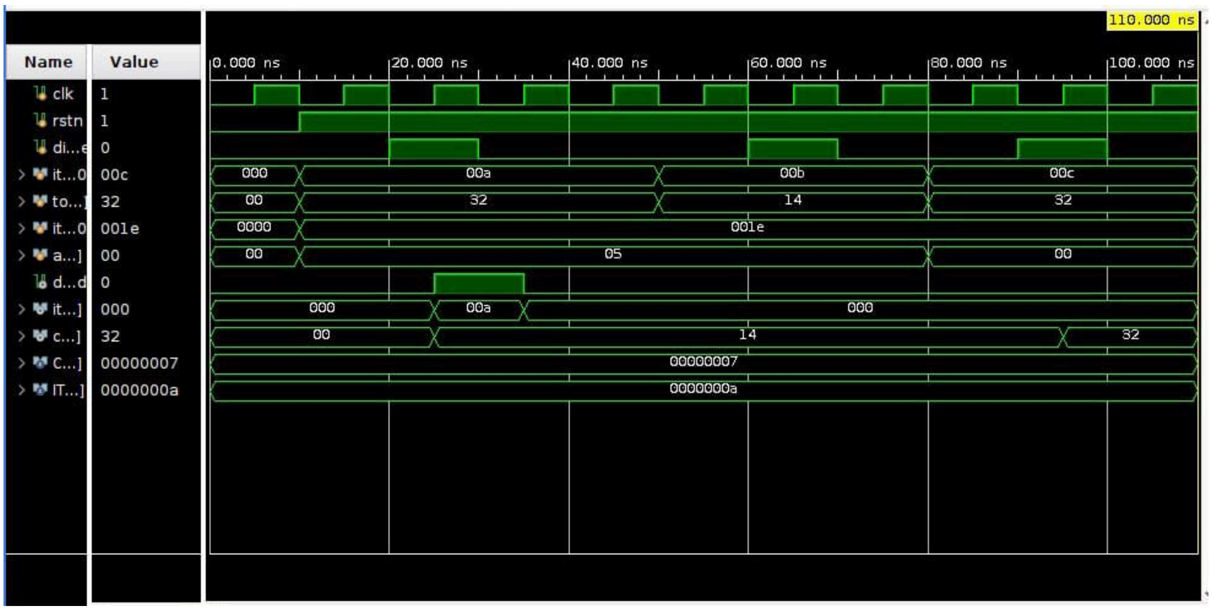


Name	Value
pclk	1
prstn	1
cfg_mode	1
psel	0
pwrite	0
paddr[14:0]	0014
pwdata[31:0]	00000000
prdata[31:0]	ZZZZZZZ
pready	0
mem_we	0
mem_waddr[9:	004
mem_wd...31:0]	a5a55a5a
mem_raddr[9:0	004
mem_rdata[31:	a5a55a5a
memory[...][31:	aaaabbbb
write_da...][31:0	aaaabbbb
my_addr[32:0]	00000001
MAX_ITE...[31:0	00000400

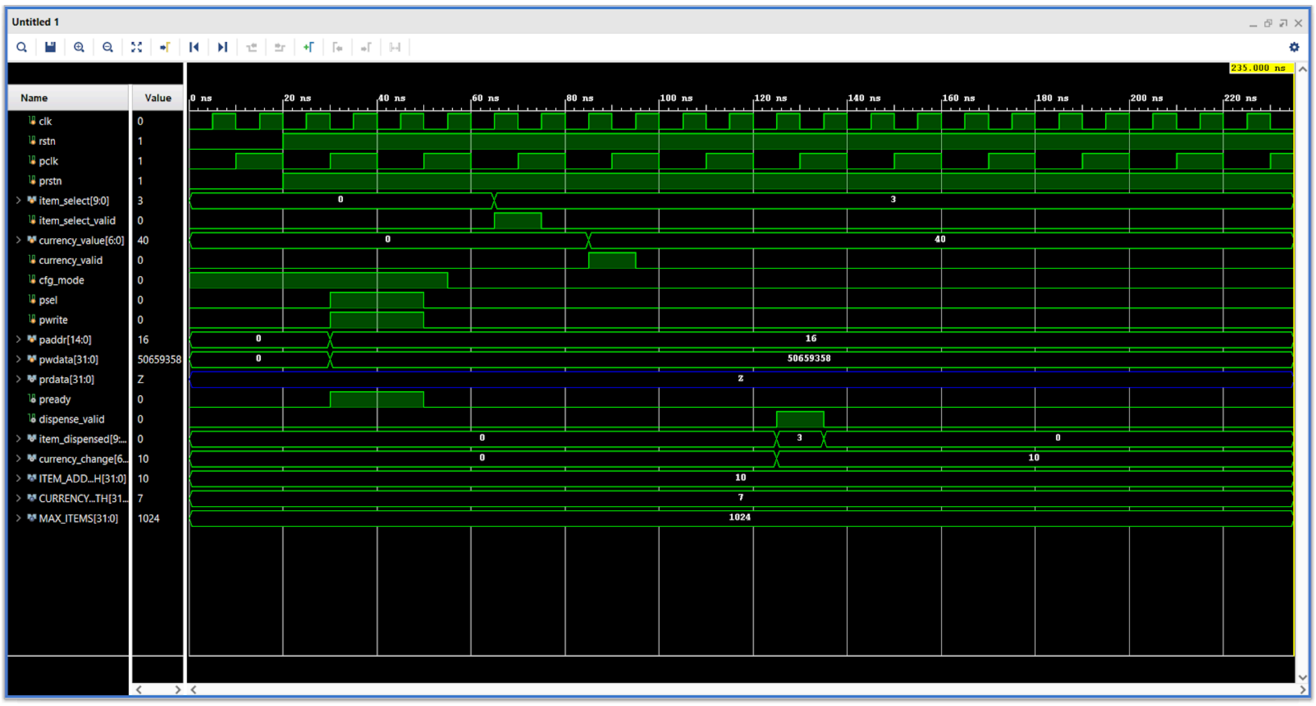
4.Main_Controller



5.Output



6.Top Module



Untitled 1

235.000 ns

Name	Value
clk	0
rstn	1
pclk	1
prstn	1
item_select[9:0]	3
item_select_valid	0
currency_value[6:0]	40
currency_valid	0
cfg_mode	0
psel	0
pwrite	0
paddr[14:0]	16
pwdata[31:0]	50659358
prdata[31:0]	Z
pready	0
dispense_valid	0
item_dispensed[9:...	0
currency_change[6...	10
ITEM_ADD...H[31:0]	10
CURRENCY...TH[31...	7
MAX_ITEMS[31:0]	1024

GithubLink:https://github.com/sure-trust/DEGA-SUSRUTHA-g4-integrated-vlsi/tree/main/Final%20capstone%20project

Learning and Reflection

Completing our Vending Machine Controller project using Verilog was not just a technical achievement it was also a meaningful learning journey for everyone on the team. As we moved through each phase designing, coding, simulating, debugging, and finally integrating the system.We faced various challenges that helped us grow both individually and as a team. These experiences deepened our understanding of digital design principles, hardware description languages, and the crucial role of synchronization across multiple clock domains.

Beyond the technical aspects, we also honed key soft skills like coordination, collaboration, time management, and clear documentation. In this section, we reflect on the specific lessons learned by each team member, highlighting both the technical knowledge gained and the overall experience of building a real-world digital system together.

Learning :

1.susrutha (TEAM LEADER):

a.Worked on the development of the Item Selection Module, which involved implementing the logic for selecting items based on user input and available stock

b. And also worked on developing the currency handling module in Verilog. This included validating user inputs, keeping track of the inserted balance, and ensuring proper coordination with the item dispensing logic.

c.Played a crucial role in bridging the members of the team, facilitating easy communication and cooperation during the project. Worked hard to solve problems and support the team, contributing significantly to the successful completion of the project.

2.Keerthi(Team member)

a. Led the top-level integration of all modules (item selection, item dispense, currency handling, APB interface) into a cohesive system.

b.Developed the Item Dispense Module, which involved implementing the logic to dispense items based on valid user input and available stock.

c. Faced challenges in ensuring communication between modules and resolving issues related to signal integrity, timing, and clock domain synchronization.

d. Worked extensively on debugging and testing the integrated system to ensure that all components worked together as expected and the system operated smoothly.

e. Learned the importance of team collaboration, as integration required close coordination with all team members to ensure the individual modules were aligned with the overall project objectives.

3. Jnaneswar (Team member)

   a) Worked on Configuration block in which the read and write operation of the data given are implemented using clock domain crossing.
   b) I learnt how to write and read the data with specific clock delays without inference with the other data.
   c) Learned how to integrate different modules in a single top module using submodules.

4. Nithin (Team Member)

   a) Contributed to the design and implementation of the Main Controller FSM, which serves as the brain of the vending machine.
   b) I understood the importance of synchronization across clock domains and I made sure the entire design followed synthesis rules, so that design is synthesizable.
   c) Through this project, I got hands-on experience in RTL design.

LEARNING EXPERIENCE :

1.Susrutha(Team Lead):

My responsibility in this project, I worked on both the **Item Selection** and **Currency Handling** modules of the vending machine controller, which gave me a strong understanding of RTL design at both the module and system levels. In the Item Selection module, I developed Verilog-based state machines to process user input and manage item availability, focusing on correct synchronization across different clock domains. I addressed challenges such as incorrect item dispensing and edge cases like item unavailability, which improved my debugging and timing analysis skills. In the Currency Handling module, I implemented logic for currency validation, balance tracking, and refund calculation. I encountered several challenges, including improper value storage and unexpected simulation behaviors like infinite loops. These issues taught me to be detail-oriented and systematic in debugging. I also gained experience selecting efficient combinational logic components like multiplexers and decoders to streamline control logic. Overall, this project enhanced my expertise in RTL coding, FSM design, timing coordination, and system integration, while also strengthening my problem-solving skills and collaborative approach in a team environment.

2.Keerthi

My main responsibility in this project included developing the **output_logic** and managing the **top-level RTL integration** of all modules. In the output_logic, I designed the logic to generate precise, one-cycle-wide dispense signals and accurately output the selected item along with the calculated change. This required careful coordination with the currency handler and control FSM to ensure timing accuracy and correct sequencing. As part of the top-level integration, I was responsible for connecting all submodules including item selection, APB interface, dual-port RAM, and output logic into a unified design. This role required me to ensure seamless communication between modules, resolve signal mismatches, and manage timing and clock domain synchronization. I collaborated closely with the team to debug system-level issues, verify integration correctness, and ensure each module functioned reliably within the overall design. This experience strengthened my skills in timing coordination, signal interfacing, and integration-level RTL design.

3. Jnaneswar

I worked on the **configuration block** in which write and read operations of the data of the item_selected are done. I have faced some problems while reading the data from the address. But, I solved the problem with the guidance of satish sir. And I have learnt using APB protocol using Clock Domain Crossing in realtime projects. And learnt how to integrate different modules and make them work one after the other to get the desired output.

4. Nithin

As a team member, I was responsible for designing the Main Controller FSM, which acts as the controller of the vending machine and controls when an item can be dispensed based on item selection and currency availability. I used the Vivado simulation tool and EDA Playground to run tests and debug timing issues, waveform mismatches, and logic errors. Through this project, I learned how to handle clock domain synchronization. I also made sure the entire design was synthesizable. This project helped me improve my understanding of FSMs, Verilog coding, simulation, debugging, and how to integrate and test a complete design.

From day one, this internship has helped me sharpen my technical knowledge and greatly improved my discipline. I can confidently say that I've learned a lot here, and I especially want to highlight how well SystemVerilog and UVM were taught. Simply thanking the trainers doesn't feel like enough the way they shared their knowledge with us is truly a noble act. Sharing knowledge is a good deed, and that's exactly what is being done here at SURE Trust."

Conclusion and Future Scope

Objectives :

The goal of the Vending Machine Controller project was to design a digital IP that defines the architecture, interfaces, and behavior of a smart controller used in vending machines. The key objectives were as follows:

1. Develop a reusable and configurable IP that can be easily integrated into different vending systems, supporting flexible item configurations.

2. Enable support for a wide range of items—up to 1024 different products—and accept multiple currency denominations including ₹5, ₹10, ₹15, ₹20, ₹50, and ₹100.

3. Ensure quick and efficient operation, with a response time of less than 10 system clock cycles from the moment valid currency is received to when the item is dispensed.

4. Implement **three distinct operating modes**:
   - **Reset Mode**: Initializes the system and sets all values to default.
   - **Configuration Mode**: Allows the machine to be loaded with item data and counters to be reset using an APB interface.
   - **Operation Mode**: Handles real-time user actions like selecting an item, inserting currency, and dispensing the product or change.

5. Standardize all interfaces for smooth communication between modules, including:
   - System clock and reset controls
   - Inputs for currency and item selection
   - Outputs for dispensing items and returning change
   - APB interface for configuration and monitoring
6. Track inventory accurately, by maintaining:
   - The current available count of each item
   - The total number of items dispensed per type

7. Provide a user-friendly experience by:
   - Instantly returning inserted currency if a selected item is out of stock
   - Clearly signaling when an item is dispensed or when a chosen item is unavailable

8. Allow easy customization through parameterization, such as:
   - Defining the total number of supported items
   - Setting the maximum accepted currency value

Achievements:

1. Reusable and Scalable Design
   I developed a flexible and reusable digital controller IP, capable of handling up to 1024 different items and supporting currency values up to ₹100. The design is fully parameterized to adapt to various vending machine configurations.

2. Support for Multiple Operating Modes.The controller works in three well-defined modes:

- Reset Mode to initialize the system,
- Configuration Mode for vendors to load items and set prices using the APB interface,
- Operation Mode where real-time user transactions like item selection and currency input take place.

3. Handling Asynchronous Currency Inputs
   The system is designed to accept currency inputs at slower, human-operated speeds (from 10KHz to 50MHz), while still operating on a 100MHz system clock. This ensures smooth handling of inputs regardless of timing differences.

4. Smart Inventory and Transaction Management
   The controller keeps track of important transaction metrics, such as the number of items dispensed, the current stock level of each item, and the price assigned to each product.

5. Real-Time Error Detection and Response
   Built-in mechanisms ensure a smooth user experience by automatically returning inserted currency if the selected item is out of stock, and clearly signaling when an item is unavailable.

6. Easy Configuration with APB Protocol
   The design includes an APB interface running at 10MHz, which allows for seamless integration with host systems and makes it easy to load item data and monitor machine status during configuration mode.

Conclusion :

The Vending Machine Controller is a flexible and well-structured digital IP core built to support modern vending machine applications. It handles real-time transactions efficiently with fast, accurate responses,thanks to its support for asynchronous currency input and customizable item counts and pricing. The design operates through clearly defined modes for reset, configuration, and normal operation, which makes it easier to maintain and integrate into larger systems. With an APB-based configuration interface, it's simple to set up and tailor to different use cases. Overall, this controller improves system performance and offers a strong platform for future upgrades in vending automation.

Future Scope :

Vending machines are no longer just simple snack dispensers—they are evolving into intelligent, feature-rich systems designed to improve convenience, support sustainability, and embrace emerging technologies. The future of vending is focused on smarter, greener, and more user-friendly experiences. Here are some key innovations shaping that future:

1. **Blockchain for Secure Transactions**
   By integrating blockchain technology, vending machines can ensure secure, transparent transactions. Every purchase can be recorded on a decentralized ledger, reducing the risk of fraud and giving customers the ability to track and verify their purchase history.
2. **Solar-Powered Machines for Sustainable Operation**
   To promote environmental responsibility, vending machines can be powered by solar energy. This is especially useful for outdoor settings or remote locations without reliable electricity, making them ideal for eco-friendly businesses or off-grid areas.
3. **AI-Powered Personalized Suggestions**
   Using artificial intelligence and machine learning, vending machines can analyze customer behavior and preferences to suggest products tailored to individual users. They could even adapt recommendations based on factors like time of day or weather, creating a more personalized experience.
4. **Touchless Interfaces with Voice and Gesture Control**
   For better accessibility and hygiene, future vending machines may use voice commands or gesture recognition. This hands-free interaction is especially valuable in public places, hospitals, or in scenarios where minimizing contact is important such as in the post-pandemic world.
5. **Recycling with Rewards**
   To encourage sustainability, vending machines could accept used packaging like bottles or cans and offer incentives in return such as discounts or product credits. This approach not only supports recycling efforts but also adds value for customers.