Experiment 3: Design of a vending machine

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | 邹易航 | **Class** | AI1 |
| **Date** | 2024.12.11 | **Score** |  |
| **Examiner** |  |  |  |

**I. Experiment procedure**

**1.1 Design a vending machine using VHDL**

The machine sells a ticket at the price of 4 yuan, and it can take either 1-yuan cash or 5-yuan cash as the input. Below is the description of the vending machine:

(1) The vending machine takes 1-yuan cash and 5-yuan cash as inputs, and generate ready, cash, output\_ticket, give\_charge and Retmoney signals. Ready = ‘1’ means the last purchase is finished and the machine is ready to start a new purchase.

(2) When reset, the machine holds in the ready state, where signal Ready = ‘1’ and other output signals are ‘0’ (LOW).

(3) When the machine is ready and it takes a 5-yuan cash, it will output the ticket and give charge in the next state (corresponding signals change to HIGH). During this process, the machine is not ready.

(4) When the machine is ready and it takes a 1-yuan cash, signal Cash = ‘1’ and the machine will keep waiting for the following 1-yuan cash until 4 1-yuan cashes have been given the machine, which allows the machine to give out a ticket in the next state. During this process, the machine is not ready.

(5) If a 5-yuan cash is given the machine when signal Cash = ‘1’, the machine will return all the money in the next state without giving out a ticket.

(6) After the purchase in (3) to (5), the machine will return to the ready state in the next clock cycle.

The vending machine design described in the problem operates with two types of cash inputs (1-yuan and 5-yuan) and has a set of output signals to manage its operations. Initially, the machine is in a "ready" state, where it can accept payments. Upon receiving a 5-yuan note, the machine immediately outputs a ticket and returns any excess charge, transitioning to a "not ready" state. If the machine receives a 1-yuan note, it accumulates these until it reaches 4 yuan, at which point it issues a ticket and enters the "not ready" state. However, if a 5-yuan note is inserted while the machine is in the middle of accumulating 1-yuan notes, it returns the money without issuing a ticket. After each transaction (whether completed or interrupted), the machine returns to the "ready" state in the next clock cycle. The VHDL design will need to account for these state transitions, handling both 1-yuan and 5-yuan inputs appropriately, and generating the corresponding output signals for ticket issuance and money return.

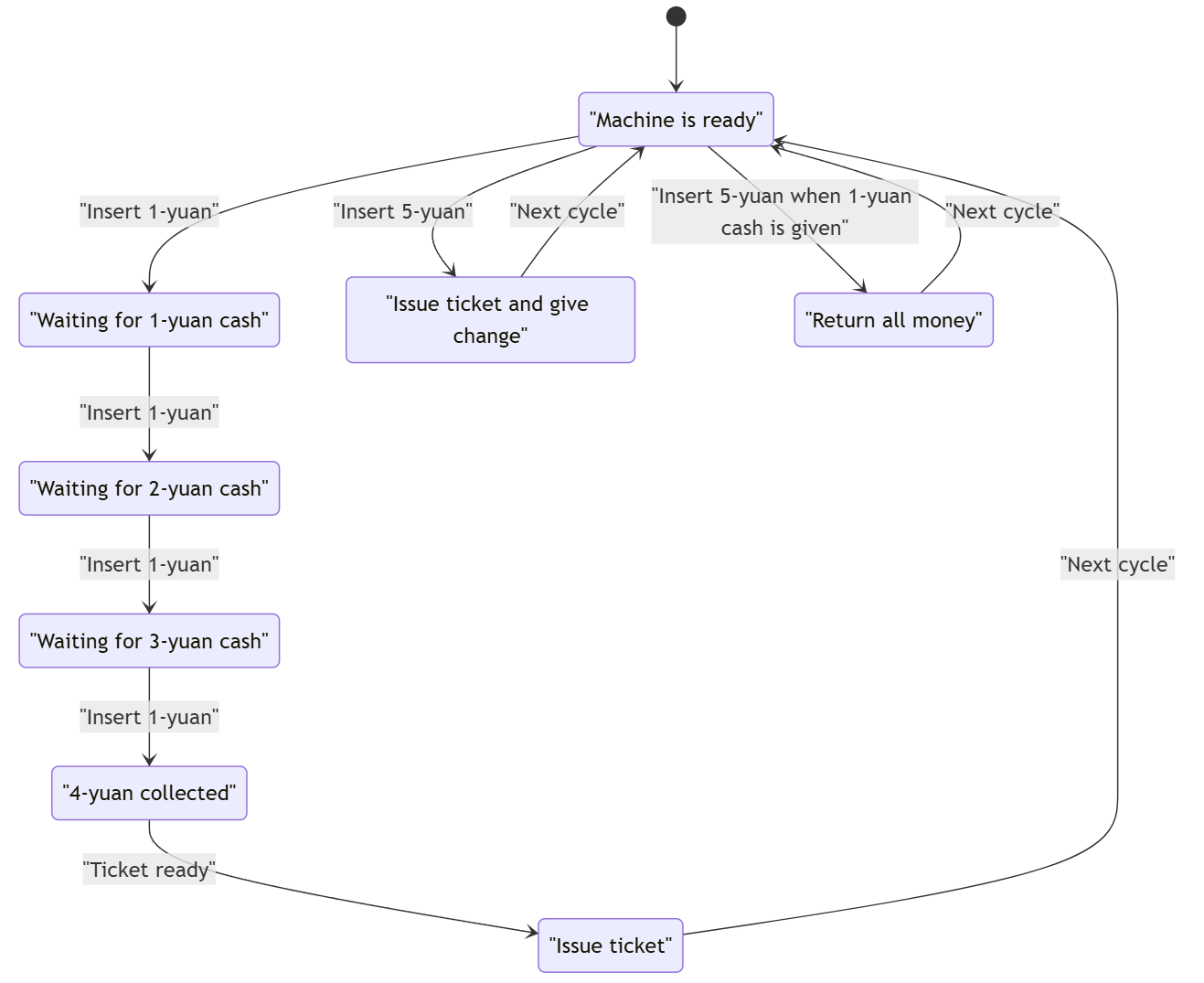


Figure 1: The diagram of the state machine structure for the vending machine

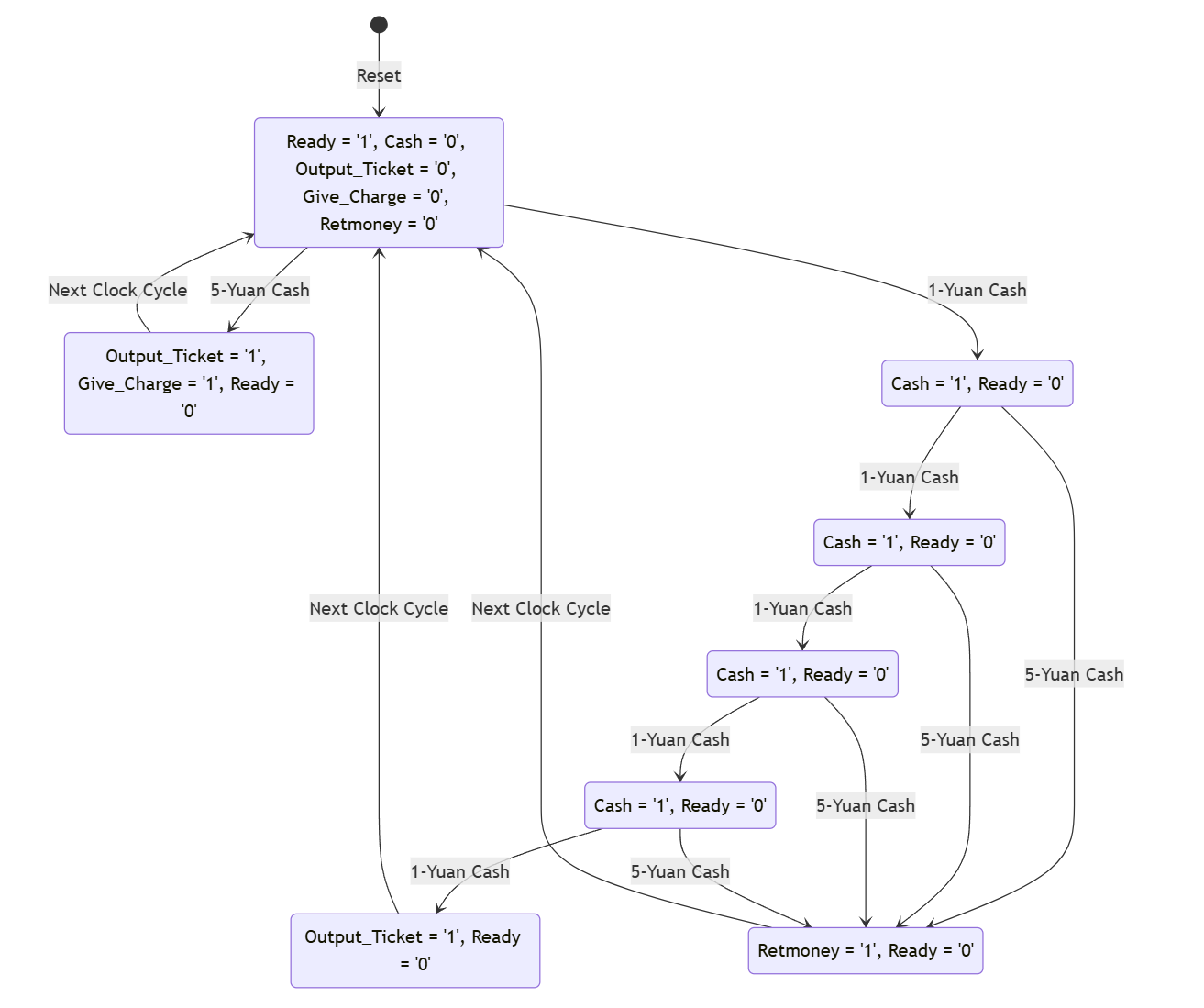


Figure 2: The state diagram of the vending machine

**1.2 Design a testbench to test the functions of the vending machine**

Design a testbench to test the functions of the vending machine by:

(1) Give the machine a 5-yuan cash when it is ready.

(2) Give the machine four 1-yuan cash when it is ready.

(3) Give the machine two 1-yuan cash and then a 5-yuan cash when it is ready.

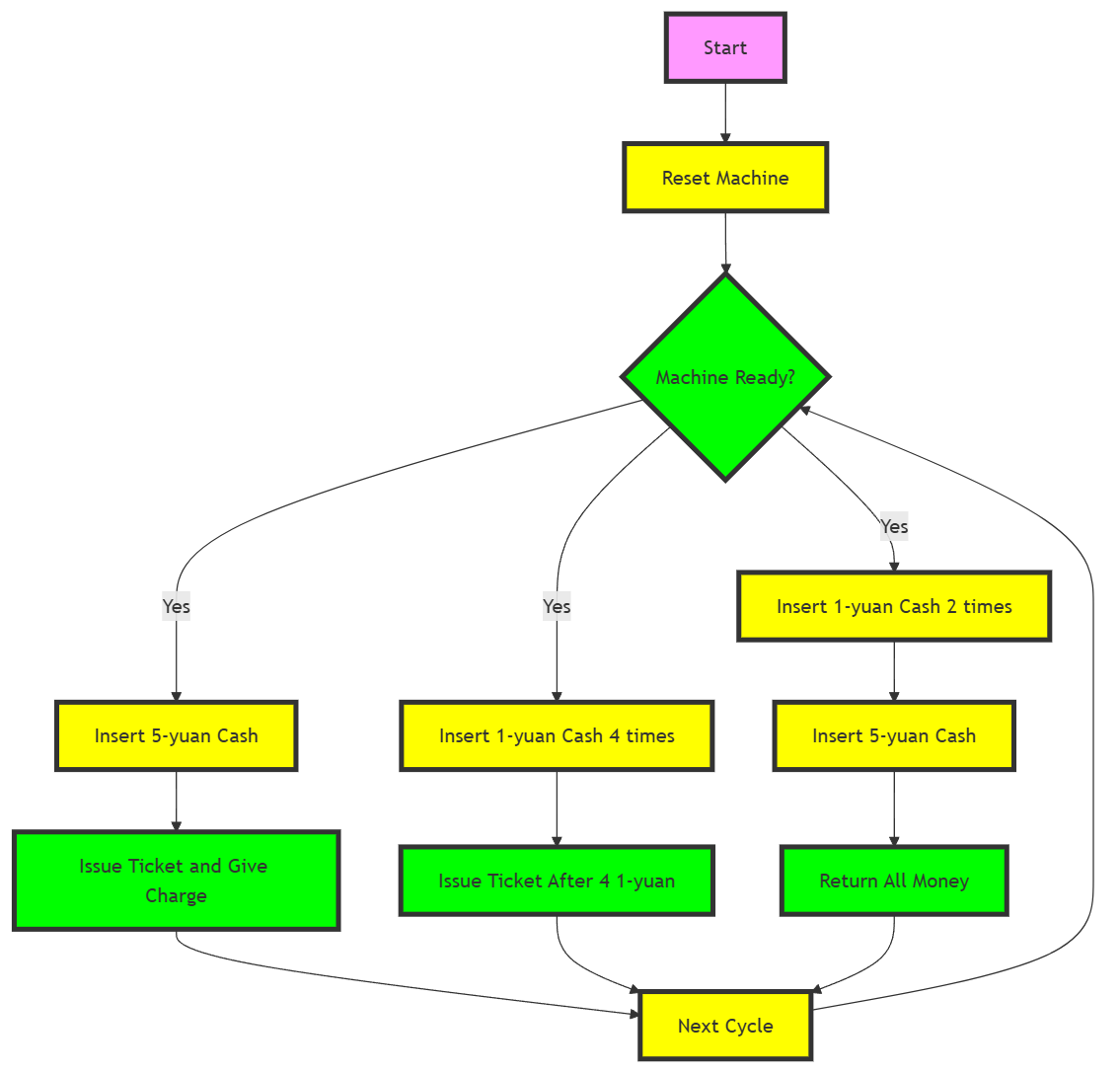


Figure 3: The diagram for the input flow of the vending machine testbench

The testbench for the vending machine should simulate the sequence of inputs as described in the problem and check the functionality of the vending machine. The testbench will provide stimulus to the machine and verify its output in each scenario. Below is a brief analysis of the three cases:

1. Test Case 1: Give the machine a 5-yuan cash when it is ready.

Expected Behavior: The machine should immediately issue a ticket and return the change (1 yuan). The machine will then transition to the "not ready" state and, in the next clock cycle, return to the "ready" state.

1. Test Case 2: Give the machine four 1-yuan cash when it is ready.

Expected Behavior: The machine will accumulate the 1-yuan notes and, after receiving the fourth one, issue a ticket. The machine should transition to the "not ready" state and return to "ready" in the next clock cycle.

1. Test Case 3: Give the machine two 1-yuan cash and then a 5-yuan cash when it is ready.

Expected Behavior: After receiving two 1-yuan notes, the machine will be expecting two more to issue a ticket. However, upon receiving a 5-yuan note, the machine should return the total accumulated amount (2 yuan) and not issue a ticket. It should then return to the "ready" state after the next clock cycle.

**II. Experimental record**

**2.1 Design the vending machine using VHDL**

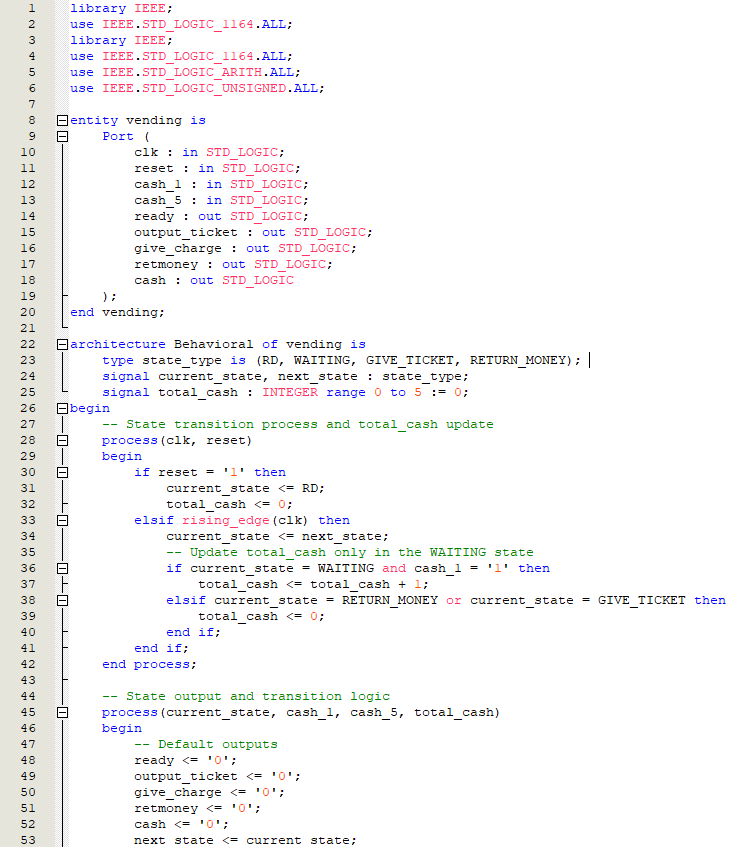


Figure 4: The VHDL code of vending machine (part 1)

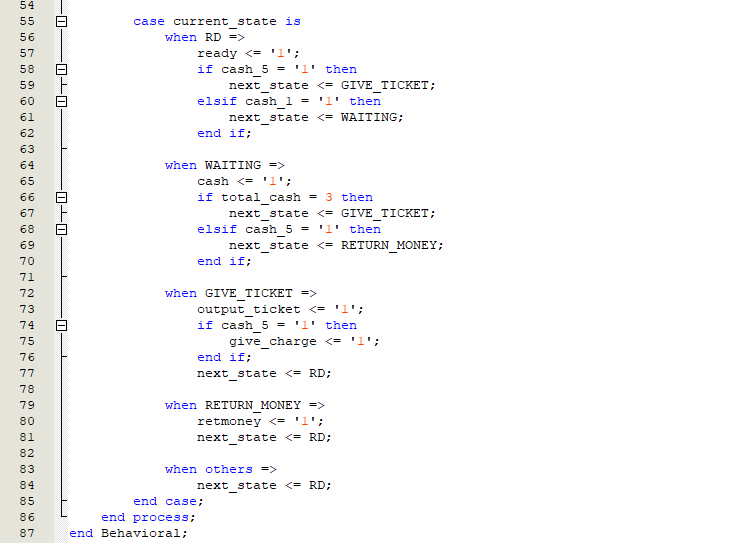


Figure 5: The VHDL code of vending machine (part 2)

The VHDL code for the vending machine comprises several components and logic blocks that collectively define its functionality. Each section of the code contributes to handling different aspects of the vending machine's operation, from initializing and setting states to managing inputs and controlling outputs based on those states.

**Entity and Signal Declaration:** In the first part of the code, an entity named vending is defined, which includes inputs and outputs that facilitate the operations of the vending machine. Inputs include clk for the clock signal, reset to initialize or reset the machine, cash\_1 and cash\_5 for monetary inputs, and outputs such as ready, output\_ticket, give\_charge, and retmoney that signal the state of the machine and the results of transactions. This setup effectively prepares the vending machine to receive inputs and produce outputs that align with its operational logic.

**Architecture and State Initialization:** Following the entity declaration, the architecture labeled Behavioral is introduced, outlining the core functionality of the vending machine. Here, a type state\_type is defined, enumerating the states (RD, WAITING, GIVE\_TICKET, RETURN\_MONEY) that the vending machine can be in. Signals like current\_state, next\_state, and total\_cash are initialized to manage transitions and track cash input. The initial state and zeroing of total\_cash when the machine is reset ensure that it starts in a predictable manner, ready to process transactions.

**State Transition Process:** A critical process block is dedicated to handling state transitions and updating total\_cash. This process is sensitive to the clk and reset signals, ensuring that state changes occur synchronously with the clock and can be reset promptly. In this block, logic to manage cash input, either accumulating it or resetting the total based on the machine's current state and input type, is implemented. The decision structure within this block directs the flow of operations, such as waiting for the correct amount of money to be inserted or handling unexpected inputs by transitioning to appropriate states like RETURN\_MONEY.

**Output Management and Detailed State Logic:** The final part of the code involves a detailed case statement that controls the outputs based on the current\_state. This section effectively decides how the vending machine should react in each state: issuing tickets, returning money, or waiting for more input. Outputs like output\_ticket, give\_charge, and retmoney are set here, ensuring that users receive appropriate feedback and actions based on their interactions with the machine. This logic not only handles the direct outputs but also guides the machine back to the RD state, ready for the next transaction after an action is completed.

This organized approach in the VHDL code ensures that each aspect of the vending machine's functionality is carefully managed, from initialization and input handling to state transitions and output control, creating a robust and user-responsive system.

**2.2 Design a testbench to test the functions of the vending machine**

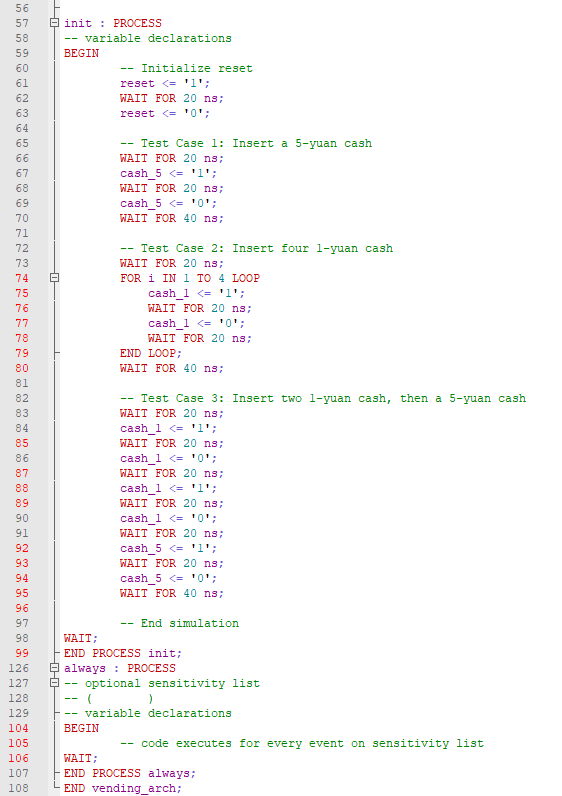
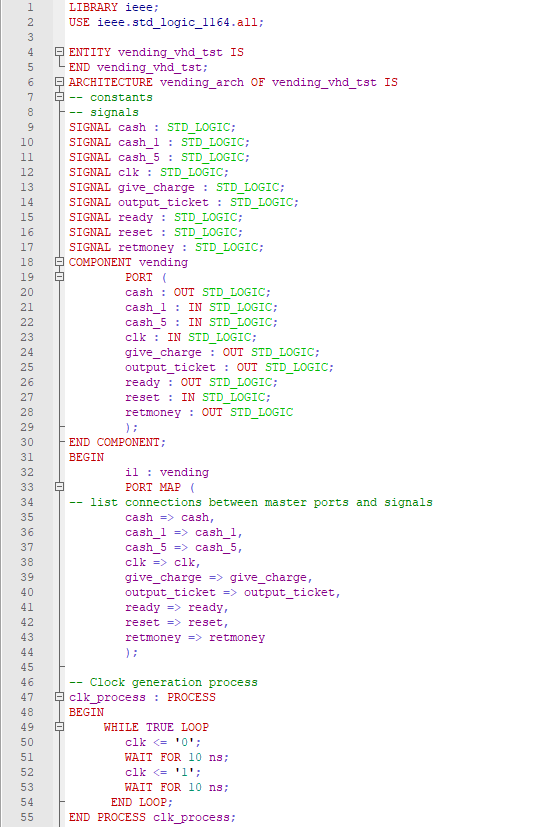


Figure 6: The test bench code for the vending machine

The VHDL testbench code provided simulates a vending machine system by establishing a test environment where the vending machine's behavior can be validated against specific cash inputs and operational scenarios. This testbench is crucial for ensuring that the vending machine functions correctly as designed, by methodically checking its response to various inputs and states.

**Entity and Architecture Setup**

* The testbench defines an entity called vending\_vhd\_tst with no external ports because it's designed to run in a simulation environment.
* The architecture, vending\_arch, contains signal declarations for interfacing with the vending machine component. These signals mirror the vending machine's ports, including inputs like cash\_1, cash\_5, reset, clk (clock), and outputs such as ready, output\_ticket, give\_charge, retmoney.

**Component Instantiation**

* Within the architecture, a component instance of the vending machine (vending) is declared and mapped to the testbench signals. This setup allows the testbench to drive the vending machine's inputs and observe its outputs directly.

**Clock Generation**

* The testbench includes a clock generation process that continuously toggles the clk signal every 10 ns, creating a 50 MHz clock. This simulated clock is essential for testing the synchronous behavior of the vending machine, which relies on clock edges to trigger state transitions.

**Initialization and Test Cases**

* A separate process named init is used for initializing the machine and running test cases:
  + **Reset Pulse**: Initially, the reset signal is pulsed high for 20 ns and then set low, which should set the vending machine to its initial state, ready to accept inputs.
  + **Test Case 1**: Tests the insertion of a 5-yuan note. The vending machine should, ideally, issue a ticket and give 1 yuan as change if the ticket price is 4 yuan.
  + **Test Case 2**: Involves inserting four 1-yuan notes consecutively. After the fourth note, the vending machine should issue a ticket, assuming the ticket costs exactly 4 yuan.
  + **Test Case 3**: Tests the machine's handling of an error condition by inserting two 1-yuan notes followed by a 5-yuan note. The expected behavior would be for the machine to return all money since the transaction sequence is incorrect for standard operation.

**Simulation Control**

* After executing the test scenarios, the simulation is paused, which typically marks the end of the testbench's active simulation phase. This is done using a simple WAIT; command, which effectively halts further simulation activities, allowing examination of the final states and outputs.

This testbench is structured to methodically verify each operational path of the vending machine, ensuring it correctly handles cash inputs, manages state transitions appropriately, and outputs correct signals based on different scenarios. This approach helps in identifying any functional anomalies and ensures the reliability of the vending machine under typical usage conditions.

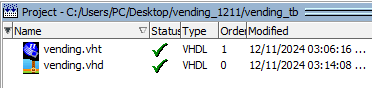


Figure 7: All vending machine VHT and VHD codes being compiled

After all codes being compiled, we can add nodes to the test bench and start simulation.

**2.3 Simulation results of the vending machine**

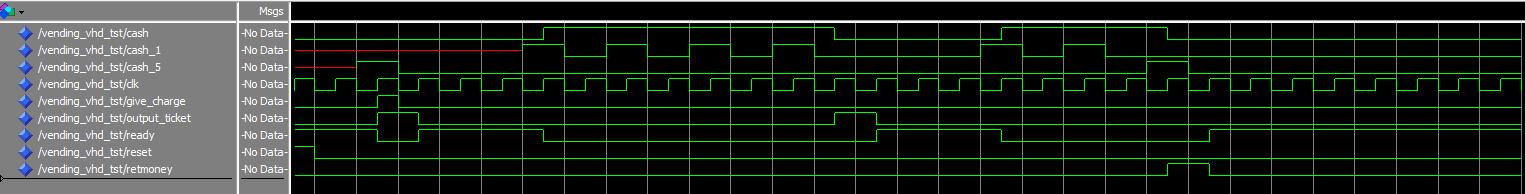


Figure 8: The simulation result of the vending machine

From the simulation results displayed, we can assess the functionality and response of the vending machine to different cash inputs as described in your test cases. The waveform indicates various signals over time, specifically showing the behavior of cash inputs (cash\_1, cash\_5), and the vending machine's outputs (give\_charge, output\_ticket, retmoney), along with the ready signal status. Let's analyze these in relation to the provided scenarios:

**Signal Description and Expectations**

* **cash\_1 and cash\_5**: Inputs for 1-yuan and 5-yuan cash respectively.
* **ready**: Indicates the machine is ready for a new transaction.
* **output\_ticket**: Indicates a ticket has been dispensed.
* **give\_charge**: Indicates change has been given back.
* **retmoney**: Indicates money is being returned without ticket issuance.

**Analysis of Simulation Results**

1. **Test Case 1: Give the machine a 5-yuan cash when it is ready.**
   * The ready signal starts high, indicating readiness.
   * Upon the insertion of a 5-yuan cash (cash\_5 goes high), the subsequent outputs should ideally show output\_ticket going high (ticket issued) and give\_charge high (1 yuan change given). From the waveform, it appears that these actions occur correctly. After issuing the ticket and change, the ready signal should go high again, indicating readiness for the next transaction.
2. **Test Case 2: Give the machine four 1-yuan cash when it is ready.**
   * With the machine initially ready, four successive 1-yuan inputs are given (cash\_1 goes high four times). The machine should only issue a ticket (output\_ticket high) after the fourth 1-yuan input without giving any change (give\_charge remains low). The ready signal should resume high after the transaction completes. The waveform should confirm that no other outputs are triggered until after the fourth 1-yuan is recognized.
3. **Test Case 3: Give the machine two 1-yuan cash and then a 5-yuan cash when it is ready.**
   * Starting ready, two 1-yuan notes are inserted followed by a 5-yuan note. Here, retmoney should go high following the 5-yuan input, reflecting that all money is returned without issuing a ticket due to incorrect payment sequence. The output\_ticket should remain low throughout, and give\_charge should also stay low as no change is due. The ready signal should revert to high post-transaction, indicating that the machine handled the erroneous input correctly and reset itself.

**Conclusion**

The simulation results should corroborate the machine’s expected behavior in handling transactions as designed:

* Correctly dispensing tickets and change or returning money as appropriate.
* Resetting to a ready state post-transaction.
* Handling sequences of cash inputs as specified in the scenarios.

Each of these operations, indicated by the changes in relevant signals on the waveform, helps validate the correct functionality of the vending machine according to the given specifications. The visual verification via waveforms in a simulation is crucial for confirming that the machine’s logic and state transitions are implemented correctly before deploying the hardware.

**2.4 The RTL block view of the vending machine**

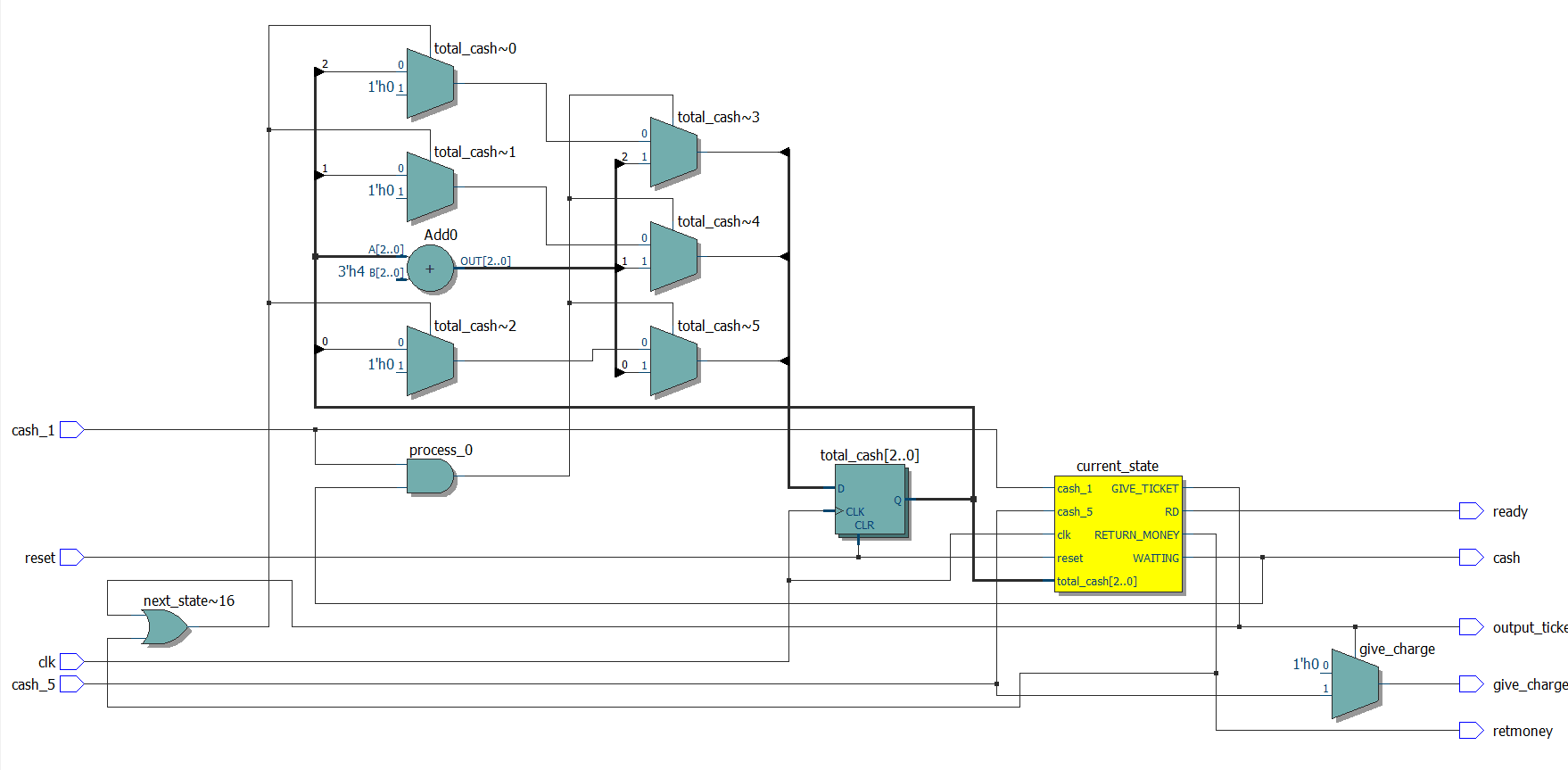


Figure 9: The RTL view of the vending machine

The diagram you provided depicts the architecture of a vending machine's control system, designed to process transactions based on cash inputs. This system is structured around a central state machine, which manages transitions based on input signals such as cash\_1 for 1-yuan coins, cash\_5 for 5-yuan notes, and a reset signal for reinitializing the machine to its base state. The clk signal ensures that these transitions occur in sync with the system clock, maintaining orderly operation throughout.

At the heart of this design is the process\_0 block, which appears to be the main decision-making component. This block evaluates the accumulated cash through a series of adders that increment each time a new coin is inserted, helping to track the total amount inserted. Based on this total and the inputs received, it determines the machine's next state—whether to continue accepting more cash, issue a ticket, give change, or return money.

Output signals such as ready, cash, output\_ticket, give\_charge, and retmoney are directly controlled by the state transitions within process\_0. The ready signal indicates when the machine is prepared to initiate a new transaction, resetting to high after each completed or reset transaction. The output\_ticket signal is triggered to dispense a ticket once the correct amount has been inserted, while give\_charge and retmoney manage the return of funds either as change due or when an invalid payment is detected.

Overall, this system is meticulously designed to ensure that every possible scenario—from accepting payments to handling errors—is managed efficiently. Each component and pathway in the diagram is crucial for ensuring that the vending machine operates reliably, providing necessary services like issuing tickets and managing cash while maintaining user satisfaction and operational integrity.

**III. Analysis and discussion**

The task of designing and analyzing a vending machine system using a VHDL-based state machine has involved several complex steps, from initial concept and entity definition to the implementation of control logic and simulation testing. The goal was to create a system that could accurately handle multiple inputs (1-yuan and 5-yuan cash) and perform specific outputs such as dispensing tickets, giving change, and returning money, all dictated by the logic defined within the VHDL architecture.

In the process, we began by outlining the essential features and operations of the vending machine, translating these requirements into a functional digital system. This included defining the inputs and outputs necessary for the machine’s operation and considering the various states the machine could occupy—such as ready, waiting for more cash, issuing a ticket, giving change, and returning money. Each state was meticulously planned to ensure that transitions occurred logically and efficiently based on the user interactions and internal conditions.

Next, we crafted the VHDL architecture, which involved setting up the entity to interface with the external signals and constructing the internal logic to manage state transitions and output responses. This step was crucial for the system’s reliability and effectiveness, ensuring that each component behaved as expected under various scenarios. The use of a state machine architecture helped in clearly defining how the machine should react to different combinations of inputs, providing a robust framework for handling transactions.

The simulation phase was then employed to verify the correctness of the design. By simulating how the machine would react to inputs like inserting multiple coins or notes, we could observe whether the outputs matched the expected results, such as issuing a ticket after receiving the exact amount or returning money when incorrect amounts were entered. These tests were vital in confirming that the logic implemented in the VHDL code effectively met the operational requirements and user expectations.

The analysis of the simulation results and the overall system behavior revealed insights into the design’s strengths and potential areas for improvement. It demonstrated the system's capability to handle expected transactions smoothly and its robustness in dealing with erroneous inputs, ensuring that the machine could operate reliably in a real-world environment. However, it also highlighted the complexity involved in managing state transitions and dealing with simultaneous inputs, pointing to the need for meticulous testing and potential refinements in the system logic.

In conclusion, this project not only involved technical design and implementation but also required a deep understanding of the operational context within which the vending machine functions. The discussion around this task illustrated the challenges and intricacies of creating a digital system that interacts dynamically with its environment, providing valuable lessons in both digital design and practical application. This endeavor showcases the critical role of detailed planning, rigorous testing, and continuous refinement in the development of effective and reliable digital systems.

**VI. Other attachments**