Computation II: embedded system design (5EIB0)

Dashboard / My courses / 5EIB0 / Practice Assessment 2024-03-08 / Part 2d: 2024-03-08 Finite State Machine (Design ATM Machine)

Description   </> Edit   Submission view

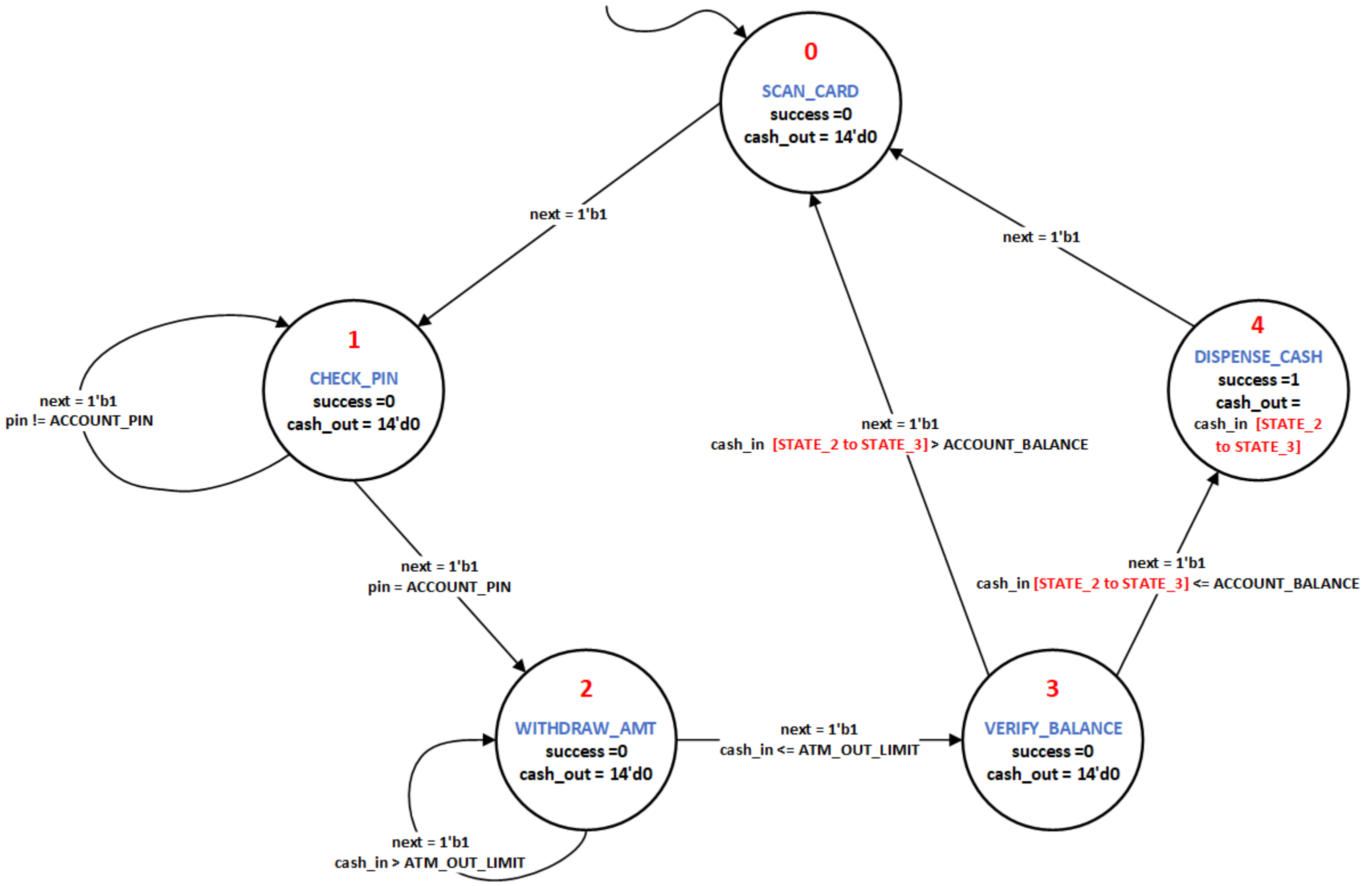
Available from: Friday, 8 March 2024, 2:30 PM
Requested files: atm_top.v (Download)
Type of work: Individual work

Introduction

In this exercise, you will create the verilog implementation of a Finite State Machine(FSM) which describes the functionality of an ATM machine. Make the following assumptions:

- The state machine has five states: **SCAN_CARD, CHECK_PIN, WITHDRAW_AMT, VERIFY_BALANCE** and **DISPENSE_CASH.**
- The ATM machine has a maximum transaction limit (**ATM_OUT_LIMIT**) of **€7000** per transaction.
- The pin of your account (**ACCOUNT_PIN**) is **1234**. Also, your current account balance (**ACCOUNT_BALANCE**) is **€3000.**

Figure below illustrates the described vending machine.



The diagram above illustrates the transitions of the FSM that should be implemented to achieve this. It shows a Moore Machine where the output is defined by the state. The machine has five states. The state numbers are marked in red.

Assignment

Write a FSM module named **atm_top** which takes five input signals **clk (1 bit), cancel (1 bit), next (1 bit), pin (16 bits), cash_in (14 bits).**

HINT1: If PIN is **1234**, then **pin** input is encoded as follows:

$$\underbrace{0001}_{1}\ \underbrace{0010}_{2}\ \underbrace{0011}_{3}\ \underbrace{0100}_{4}$$

HINT2: If cash required is **€50**, then **cash_in** input is encoded as follows: **00000000110010 (50 in binary)**.

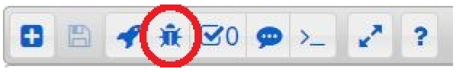
The state machine has three output signals **success(1 bit), cash_out(14 bits), state_display(3 bits)**. All state transition take place at **posedge** of **clk**. The **cancel** input always makes a transition to the **SCAN_CARD** state. **state_display** outputs the current state.

NOTE1: Make sure that you only make one transition every time **next** is set to high. For doing this check whether the current value is different from its previous value. Do not use: always @ (posedge **next**).

NOTE2: Ideally, in an ATM machine, one is not allowed to change the cash withdrawal amount unless the transaction finishes or the transaction is cancelled. So, you need to store the value of the **cash_in** given during transition to state **VERIFY_BALANCE([STATE_3])** from state **WITHDRAW_AMT ([STATE_2])** for further verification in **VERIFY_BALANCE** and cash dispensing in **DISPENSE_CASH**.

Debug

Click on the symbol marked below to see the waveforms produced by your design. Please note that if your code has an error that prevents it being simulated it will not produce any waveforms.



VPL