|  |
| --- |
| ELEC 402 |
| Project 1 Report |
| Finite State Machines; https://github.com/mchuahua/ELEC402/tree/master/Proj1 |

|  |
| --- |
| Martin Chua - 35713411  9-29-2021 |

**General Description**

The Finite State Machine (FSM) is of a generic bank ATM for withdrawing and depositing money. The FSM is intended to be instantiated with parameter settings for CORRECT\_PIN, SAVINGS\_FUNDS\_AMOUNT, and CHEQUING\_FUNDS\_AMOUNT, which are then stored locally in the instantiation following a cycle with reset high. If left default, the pin will be 1234, savings will contain 1000, and chequing will contain 25. States consist of 4 major sections with a total of 13 states:

* Initial phase (Initial startup, pin validation, and selecting deposit/withdrawals)
* Deposit phase (Account selection, depositing cash vs check, open atm deposit slot)
* Withdrawal phase (Account selection, withdraw amount, checking for insufficient funds, open atm withdrawal slot)
* End phase (Withdrawal of card)

The states are controlled based on inputs to the module and outputs allow for ATM to signal ready, open atm out (deposit), or open atm in (withdrawal). For more information on the states, see below sections.

As this is a basic bank ATM FSM to demonstrate FSM rather than bank ATM functionality, there may be some differences in the bank ATM FSM function compared to ones in real life. I.e. this bank ATM FSM is not realistic…

**IO + FSM Modules + Testbench**

Input/outputs definitions and the purpose/description of each state are below. Unless otherwise specified, inputs/outputs are one bit width in size.

Testbench is commented within the code as specified within the project documentation.

FSM Inputs

* clk
  + Basic clock to drive entire FSM module
* rst
  + Basic reset to initialize entire FSM module, and to reset it if anything happens
* bank\_card\_insert
  + Signal to indicate that a bank card has been inserted, thus starting the various FSM states
* deposit\_withdrawal\_selection
  + Signal to select a deposit or withdrawal, indicated by 0 – Withdrawal, 1 – Deposit
* account\_selection
  + Signal to select account, either 0 – Chequing, 1 – Savings
* amount[13:0]
  + Bus to indicate amount for withdrawing or depositing.
* pin[13:0]
  + Bus to indicate input pin, there is validation for correct pin vs input pin.

FSM outputs

* open\_atm\_out
  + Signal to open the ATM deposit out slot for dispensing cash
* open\_atm\_in
  + Signal to open the ATM withdrawal in slot for receiving cash or check
* ready
  + Signal to indicate that the ATM is ready to be used (and not used by another)

States

The 13 states are initialized using “enum”, which allows for ease of reading during test-benching (as seen in the waveforms in the last part). The states are as follows:

* idle
  + (1) Idle state for after reset.
* pin\_check
  + (2) Pin check state for validating input pin is correct to local pin.
* select\_deposit\_withdrawal
  + (3) Deposit/Withdrawal check state for selecting either deposit or withdrawal
* deposit\_account\_selection
  + (4) If deposit was chosen in (3), this state starts the deposit phase and selects the deposit account
* deposit\_cash\_or\_check
  + (5) Confirmation state that moves to open\_atm\_in state
* open\_atm\_in
  + (6) State for ATM deposit slot to be opened.
* withdrawal\_account\_selection
  + (7) If withdrawal was chosen in (3), this state starts the deposit phase and selects the withdrawal account
* withdrawal\_amount\_selection
  + (8) Records amount to be withdrawn from the amount input
* withdraw\_chequing
  + (9) Confirmation state for chequing (automatically moves to insufficient funds check)
* withdraw\_savings
  + (10) Confirmation state for savings (automatically moves to insufficient funds check)
* insufficient\_funds\_check
  + (11) Checks for insufficient funds in chosen chequing/savings to withdraw from
* open\_atm\_out
  + (12) State for ATM withdrawal slot to be opened.
* withdraw\_card
  + (13) Loop to make sure card is withdrawn (bank\_card\_insert set to low)

Testbench input outputs

Testbenches do not have input / outputs as testbenches are not supposed to have them. However, it does instantiate the dut, which in this case is the FSM.

Testbench tests

There are various tests implemented in the testbench. The majority of these tests verify state transition and use asserts + error counts to log how many failed. These include:

* Assert ready for when bank card isn’t inserted
* Incorrect pin test using a pin that does not match the correct pin
* Correct pin test using the correct pin for state transition
* Withdrawal selection test for selecting withdrawal or deposit
* Deposit selection (and reset) tests for correct resetting and deposit account selection
* Deposit funds amount test for the correct record of funds deposited
* Withdrawal amount test for correct record of funds withdrawn
* Insufficient funds test using an amount greater than what is present in the specified account
* Withdraw card test to make sure card is withdrawn

**FSM Block Diagram**

See .\fsm.drawio

Text

Description automatically generated

**Module + Testbench Block Diagram**

See .\fsm\_tb\_connection.drawio

**Graphical user interface, text, application

Description automatically generatedFSM State Diagram**

See .\fsm\_state\_diagram.drawio for file. Blue text indicates data being set, the black text indicates what causes the states to change.

**Diagram

Description automatically generated**

**Copy of code**

See .\code.pdf

**Simulation waveform results**

The testbench tests most possible ways of traversing the FSM. The list of tests can be seen above, or listed below to match the simulation results:

**A picture containing graphical user interface

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

**Graphical user interface, text, application

Description automatically generated**