module ATM(

input wire clk,

input wire reset,

input wire faceDetected,

input wire[3:0] keypad,

input wire[3:0] userPIN,

input wire[15:0] withdrawalAmount,

output wire[6:0] display,

output wire cashDispensed,

output wire faceUnlockRequest

);

// Define states for the ATM machine

parameter IDLE = 3'b000;

parameter AWAITING\_PIN = 3'b001;

parameter AWAITING\_FACE\_UNLOCK = 3'b010;

parameter PROCESSING\_TRANSACTION = 3'b011;

parameter DISPENSE\_CASH = 3'b100;

parameter TRANSACTION\_COMPLETE = 3'b101;

// Internal registers

reg [2:0] state;

reg [2:0] nextState;

reg [15:0] accountBalance;

reg [15:0] remainingBalance;

// Other internal signals and registers

reg [2:0] incorrectPINAttempts;

reg [2:0] incorrectFaceAttempts;

// Set initial values

initial begin

state <= IDLE;

accountBalance <= 100000; // Initial account balance

incorrectPINAttempts <= 0;

incorrectFaceAttempts <= 0;

end

// State transition and output logic

always @(posedge clk or posedge reset) begin

if (reset) begin

state <= IDLE;

accountBalance <= 100000; // Reset account balance

incorrectPINAttempts <= 0;

incorrectFaceAttempts <= 0;

end

else begin

state <= nextState;

case (state)

IDLE:

if (keypad == 4'b0001) begin // Keypad "1" pressed

nextState = AWAITING\_PIN;

end else if (keypad == 4'b0010) begin // Keypad "2" pressed

nextState = AWAITING\_FACE\_UNLOCK;

end else begin

nextState = IDLE;

end

AWAITING\_PIN:

if (userPIN == 4'b0101) begin // Correct PIN entered

nextState = PROCESSING\_TRANSACTION;

end else begin

incorrectPINAttempts <= incorrectPINAttempts + 1;

if (incorrectPINAttempts >= 3) begin

nextState = IDLE; // Lock account after 3 incorrect attempts

end else begin

nextState = AWAITING\_PIN;

end

end

AWAITING\_FACE\_UNLOCK:

if (withdrawalAmount == 16'd10000 && faceDetected) begin

nextState = PROCESSING\_TRANSACTION;

end else begin

incorrectFaceAttempts <= incorrectFaceAttempts + 1;

if (incorrectFaceAttempts >= 3) begin

nextState = IDLE; // Lock account after 3 incorrect attempts

end else begin

nextState = AWAITING\_FACE\_UNLOCK;

end

end

PROCESSING\_TRANSACTION:

if (withdrawalAmount <= accountBalance) begin

nextState = DISPENSE\_CASH;

end else begin

nextState = TRANSACTION\_COMPLETE;

end

DISPENSE\_CASH:

nextState = TRANSACTION\_COMPLETE;

TRANSACTION\_COMPLETE:

nextState = IDLE;

default:

nextState = IDLE;

endcase

end

end

// Output assignment logic

always @(state or withdrawalAmount or accountBalance or incorrectPINAttempts) begin

case (state)

// ...

DISPENSE\_CASH:

display = 7'b0001000; // Dispensing cash

cashDispensed = 1'b1; // Set to 1 when cash is being dispensed

faceUnlockRequest = 1'b0;

// ...

default:

display = 7'b0000000;

cashDispensed= 1'b0; // Set to 0 by default to indicate no cash being dispensed

faceUnlockRequest = 1'b0;

endcase

end

endmodule