module Project (

input wire clk,

input wire reset,

// User Interface module inputs

input wire touchscreen\_input,

input wire [3:0] button\_input,

// Barcode Scanner module inputs

input wire barcode\_detection,

input wire [7:0] barcode\_data,

// Cash Handling module inputs

input wire [3:0] cash\_insertion,

input wire [2:0] denomination\_selection,

input wire cash\_confirmation,

// Payment Processing module inputs

input wire [31:0] confirmed\_amount,

input wire [31:0] inserted\_cash\_amount,

input wire payment\_confirmation,

// Acknowledgment module inputs

input wire transaction\_complete,

input wire [31:0] transaction\_amount,

// User Interface module outputs

output wire touchscreen\_prompt,

output wire [3:0] button\_selection,

// Barcode Scanner module outputs

output wire voucher\_detected,

output wire [7:0] captured\_data,

// Cash Handling module outputs

output wire cash\_accepted,

output wire [3:0] inserted\_cash\_amount\_output,

// Payment Processing module outputs

output wire [31:0] adjusted\_amount,

output wire excess\_payment,

output wire short\_payment,

output wire disconnection,

// Acknowledgment module outputs

output wire acknowledgment\_issued

);

// Instantiate modules

endmodule

module UserInterface (

input wire clk,

input wire reset,

input wire touchscreen\_input,

input wire [3:0] button\_input,

output reg touchscreen\_prompt,

output reg [3:0] button\_selection

);

always @(posedge clk or posedge reset) begin

if (reset) begin

touchscreen\_prompt <= 0;

button\_selection <= 0;

end else begin

touchscreen\_prompt <= touchscreen\_input;

button\_selection <= button\_input;

end

end

endmodule

module BarcodeScanner (

input wire clk,

input wire reset,

input wire barcode\_detection,

input wire [7:0] barcode\_data,

output reg voucher\_detected,

output reg [7:0] captured\_data

);

always @(posedge clk or posedge reset) begin

if (reset) begin

voucher\_detected <= 0;

captured\_data <= 0;

end else begin

if (barcode\_detection) begin

voucher\_detected <= 1;

captured\_data <= barcode\_data;

end else begin

voucher\_detected <= 0;

captured\_data <= 0;

end

end

end

endmodule

module CashHandling (

input wire clk,

input wire reset,

input wire [3:0] cash\_insertion,

input wire [2:0] denomination\_selection,

input wire cash\_confirmation,

output reg cash\_accepted,

output reg [3:0] inserted\_cash\_amount\_output

);

reg [3:0] inserted\_cash\_amount;

always @(posedge clk or posedge reset) begin

if (reset) begin

cash\_accepted <= 0;

inserted\_cash\_amount <= 0;

end else begin

if (cash\_insertion != 0) begin

cash\_accepted <= cash\_confirmation;

inserted\_cash\_amount <= cash\_insertion;

end else begin

cash\_accepted <= 0;

inserted\_cash\_amount <= 0;

end

end

end

always @(posedge clk or posedge reset) begin

if (reset) begin

inserted\_cash\_amount\_output <= 0;

end else begin

inserted\_cash\_amount\_output <= inserted\_cash\_amount;

end

end

endmodule

module PaymentProcessing (

input wire clk,

input wire reset,

input wire [31:0] confirmed\_amount,

input wire [31:0] inserted\_cash\_amount,

input wire payment\_confirmation,

output reg [31:0] adjusted\_amount,

output reg excess\_payment,

output reg short\_payment,

output reg disconnection

);

reg [31:0] adjusted\_amt;

always @(posedge clk or posedge reset) begin

if (reset) begin

adjusted\_amount <= 0;

excess\_payment <= 0;

short\_payment <= 0;

disconnection <= 0;

adjusted\_amt <= 0;

end else begin

adjusted\_amt <= confirmed\_amount - inserted\_cash\_amount;

if (payment\_confirmation) begin

if (adjusted\_amt > 0) begin

adjusted\_amount <= adjusted\_amt;

excess\_payment <= 0;

short\_payment <= 1;

disconnection <= 0;

end else if (adjusted\_amt < 0) begin

adjusted\_amount <= 0;

excess\_payment <= 1;

short\_payment <= 0;

disconnection <= 0;

end else begin

adjusted\_amount <= 0;

excess\_payment <= 0;

short\_payment <= 0;

disconnection <= 0;

end

end else begin

adjusted\_amount <= 0;

excess\_payment <= 0;

short\_payment <= 0;

disconnection <= 1;

end

end

end

endmodule

module Acknowledgment (

input wire clk,

input wire reset,

input wire transaction\_complete,

input wire [31:0] transaction\_amount,

output reg acknowledgment\_issued

);

always @(posedge clk or posedge reset) begin

if (reset) begin

acknowledgment\_issued <= 0;

end else begin

if (transaction\_complete) begin

acknowledgment\_issued <= 1;

end else begin

acknowledgment\_issued <= 0;

end

end

end

endmodule