

ĐỀ THI THỰC HÀNH FPGA – VIẾT CODE VÀ CHẠY TRÊN VIVADO

Môn học: Lập trình Verilog cơ bản trên FPGA

Board sử dụng: EZBA4205 (Zynq-7000)

Thời gian: 120 phút

1. Mục tiêu

Sinh viên thực hành:

- Viết các module Verilog cơ bản (rotary encoder, ALU, LED 7 đoạn, controller).
 - Tích hợp thành hệ thống hoàn chỉnh.
 - Build project trên Vivado và chạy thử nghiệm trên board.
-

2. Yêu cầu chi tiết

Câu 1 – Module Rotary Encoder (2 điểm)

Đặc tả:

- Đầu vào:
 - `clk` : clock hệ thống (40 MHz).
 - `rst_n` : reset tích cực thấp.
 - `DT_A` : tín hiệu encoder kênh A (active low).
 - `CLK_B` : tín hiệu encoder kênh B (active low).
 - Đầu ra:
 - `rotary_up` : lên mức 1 khi xoay encoder theo chiều tăng.
 - `rotary_down` : lên mức 1 khi xoay encoder theo chiều giảm.
 - `rotary_clk` : xung nhịp phát ra mỗi lần xoay (theo xung 50Hz)
-

Câu 2 – Module ALU (3 điểm)

Đặc tả:

- Đầu vào:
 - `A` : số hạng A, 3 bit.
 - `B` : số hạng B, 3 bit.
 - `opcode` : mã lệnh 3 bit, chọn phép toán (ADD, SUB, AND, OR, XOR, ...).
 - `enable` : tín hiệu cho phép thực hiện.
 - Đầu ra:
 - `result` : kết quả 4 bit.
 - `ack` : cờ báo hiệu đã hoàn thành phép toán.
-

Câu 3 – Module LED 7 đoạn (2 điểm)

Đặc tả:

- Đầu vào:
 - `clk` : clock hệ thống.
 - `rst_n` : reset tích cực thấp.
 - `data_in` : giá trị 4 bit cần hiển thị (result từ ALU).
 - Đầu ra:
 - `seg[7:0]` : tín hiệu điều khiển 7 đoạn
 - `an[2:0]` : tín hiệu chọn LED (multiplex 3 LED).
 - Yêu cầu:
 - Hiển thị số theo hệ thập phân hoặc hex.
 - Áp dụng nguyên tắc quét LED (multiplexing).
-

Câu 4 – Module Controller (3 điểm)

Đặc tả:

- Đầu vào:
 - `clk, rst_n`.
 - Các tín hiệu từ rotary encoder: `rotary_up, rotary_down, rotary_clk`.
 - `rotary_button`: nút nhấn để thay đổi opcode.
 - Đầu ra:
 - Giá trị A và B (3 bit).
 - Giá trị opcode (3 bit).
 - `enable` đưa vào ALU.
 - Nhận lại `result, ack` từ ALU và truyền cho LED driver.
 - Yêu cầu:
 - Xoay encoder để tăng/giảm A hoặc B.
 - Nhấn nút để thay đổi opcode.
 - Phát `enable` để thực hiện phép toán.
-

3. Thực hành Vivado

- Tạo project mới trong Vivado, target board EZBA4205.
 - Viết code cho các module ở trên.
 - Tạo file **file top dung HDL_wrapper của vivado**
 - Viết file `.xdc` gán chân cho encoder, button, LED 7 đoạn.
 - Build project, generate bitstream và nạp xuống board.
 - Kiểm tra kết quả:
 - Xoay encoder thay đổi A hoặc B.
 - Nhấn nút đổi phép toán.
 - Kết quả hiển thị trên LED 7 đoạn.
-

4. Báo cáo (nộp file kèm)

- File Verilog .v của tất cả module.
 - File top-level module.
 - File .xdc.
 - Ảnh chụp kết quả chạy trên board.
-

5. Thang điểm

- Câu 1: 20 điểm
- Câu 2: 30 điểm
- Câu 3: 20 điểm
- Câu 4: 30 điểm
- Tổng cộng: 100 điểm.

```
/*
```

Module Rotary Encoder

Mục tiêu:

Thiết kế một module Verilog để giao tiếp với rotary encoder có 2 kênh A, B và một nút nhấn SW (tích cực thấp). Module phải giải mã được chiều xoay, phát ra xung clock ứng với chiều tăng (rotary_up) hoặc chiều giảm (rotary_down). Đồng thời có clock chia xuống 50Hz để làm tín hiệu lấy mẫu.

Tín hiệu đầu vào:

- clk_40MHz : xung clock hệ thống 40 MHz.
- rst_n : tín hiệu reset tích cực thấp.
- A : kênh A từ rotary encoder.
- B : kênh B từ rotary encoder.
- SW_n : tín hiệu nút nhấn từ rotary (active low).

Tín hiệu đầu ra:

- clk_50Hz : xung clock đã chia từ 40 MHz xuống khoảng 50 Hz, dùng để lấy mẫu tín hiệu encoder.
- rotary_up : xung clock mức cao 1 chu kỳ 50 Hz, phát ra khi xoay encoder theo chiều tăng.
- rotary_down : xung clock mức cao 1 chu kỳ 50 Hz, phát ra khi xoay encoder theo chiều giảm.
- (tùy chọn) rotary_clk : xung clock phát ra mỗi lần xoay, bất kể chiều nào (nếu cần cho ứng dụng sau).

Yêu cầu hoạt động:

- Module phải chia clock 40 MHz xuống 50 Hz để dùng làm clock xử lý, tránh nhiễu cơ khí.
- Phát hiện sự kiện xoay bằng cách lấy mẫu A, B theo clock 50 Hz.
- Mỗi lần xoay sinh ra 1 xung duy nhất ở rotary_up hoặc rotary_down (không giữ mức lâu).
- Xung ra có độ rộng đúng 1 chu kỳ 50 Hz ($\approx 20\text{ms}$).
- Nút nhấn SW_n có thể mở rộng thêm xử lý debounce nếu cần.

```
*/
```

```
module rotary_encoder (
```

```
    input wire clk_40MHz, // clock gốc 40MHz
```

```
    input wire rst_n,     // reset tích cực thấp
```

```
    input wire A,         // tín hiệu kênh A từ rotary
```

```
    input wire B,         // tín hiệu kênh B từ rotary
```

```
    input wire SW_n,      // nút nhấn (active low)
```

```
    output reg clk_50Hz,  // clock chia ra 50Hz
```

```
    output reg rotary_up, // xung clock chiều tăng
```

```
    output reg rotary_down // xung clock chiều giảm
```

```
        output reg SW
```

```
);
```

```

/*
Assigmen ACCA - Câu 2 ALU
Đầu vào:
    A, B (giá trị toán hạng)
    opcode[1:0] chọn phép toán
    EN (enable, bắt đầu tính toán)
Đầu ra:
    Result (kết quả phép toán)
    ACK (xác nhận đã tính toán xong, mức cao tích cực dương)
Phép toán:
    00 : cộng
    01 : trừ
    10 : and
    11 : or
*/
module ALU (
    input wire    clk,    // clock hệ thống
    input wire    rst_n,  // reset tích cực thấp
    input wire    EN,     // tín hiệu enable bắt đầu thực hiện
    input wire [1:0] opcode, // mã lệnh: 00-add, 01-sub, 10-and, 11-or
    input wire [3:0] A,    // toán hạng A
    input wire [3:0] B,    // toán hạng B
    output reg [4:0] Result, // kết quả (cộng/trừ cần 5 bit để tránh tràn)
    output reg     ACK      // báo tính toán xong
);

```

/*Yêu cầu Module 7SegLeds

Đầu vào (inputs)

- clk_40MHz : clock hệ thống 40 MHz
- rst_n : tín hiệu reset tích cực thấp
- bcd_opcode[2:0] : mã lệnh hiển thị trên LED thứ 3 (LED bên trái cùng)
- bcd_value[4:0] : giá trị hiển thị trên LED thứ 2 và LED thứ 1 (LED giữa và LED phải), phạm

vi 0–15

Đầu ra (outputs)

- SegLeds[7:0] : 8 bit điều khiển 7 đoạn + H (ABCDEFGH), mức 0 = sáng, 1 = tắt
- DP : điều khiển dấu chấm thập phân (decimal point)
- AN01, AN02, AN03 : chọn LED (common anode, mức thấp = chọn LED)
- AN01 = LED1 (bên phải, hàng đơn vị của bcd_value)
- AN02 = LED2 (bên giữa, hàng chục của bcd_value, hiển thị 0 hoặc 1)
- AN03 = LED3 (bên trái, hiển thị bcd_opcode)

Chức năng hiển thị

- LED1 hiển thị 4 bit thấp bcd_value[3:0] (0–F).
- LED2 hiển thị bit bcd_value[4] (0 hoặc 1).
- LED3 hiển thị bcd_opcode[2:0] (0–7).

Multiplexing / Quét LED

- Sử dụng kỹ thuật quét (multiplex) lần lượt bật AN01, AN02, AN03.
- Mỗi LED được refresh với tần số khoảng 100 Hz để mắt người nhìn thấy ổn định (tức cả 3 LED quét ở ~300 Hz).

- Clock chia từ 40 MHz để đạt tần số quét mong muốn.
- Giải mã BCD sang 7 đoạn
- Bảng mã hiển thị 0–9, A–F theo chuẩn common-anode (0 = sáng, 1 = tắt).
- Khi dữ liệu không hợp lệ thì tắt cả các đoạn tắt.
- Dấu chấm (DP)
- Luôn tắt (logic 1).
- Có thể mở rộng trong tương lai để hiển thị.

*/

module sevenseg_display (

```
input wire    clk_40MHz, // clock hệ thống 40MHz
input wire    rst_n,     // reset tích cực thấp
input wire [2:0] bcd_opcode, // opcode hiển thị ở LED 3
input wire [4:0] bcd_value, // value hiển thị ở LED 2 và 1 (0-15)
output reg [7:0] SegLeds, // mã 7 đoạn + H (ABCDEFGH)
output reg    DP,        // dấu chấm
output reg    AN01,      // chọn LED1
output reg    AN02,      // chọn LED2
output reg    AN03       // chọn LED3
```

);

```

/*
Module Controller
Mục tiêu:
Thiết kế module Controller để kết nối và điều phối hoạt động giữa Rotary Encoder, ALU, và 7-Segment Display.
Người dùng nhập số liệu A, B và opcode bằng Rotary Encoder và nút nhấn. Controller gửi dữ liệu sang ALU để
tính toán và đưa kết quả hiển thị ra LED 7 đoạn.
Tín hiệu đầu vào:
- clk_40MHz : clock hệ thống cho ALU và hiển thị.
- clk_50Hz : clock chậm từ Rotary Encoder (đã chia từ 40 MHz).
- rst_n : tín hiệu reset tích cực thấp.
- rotary_up : xung clock khi xoay rotary theo chiều tăng.
- rotary_down : xung clock khi xoay rotary theo chiều giảm.
- sw_pressed : xung nhấn nút (đã debounce).
- ack : tín hiệu báo ALU tính toán xong.
- alu_result[3:0] : kết quả tính toán từ ALU.
Tín hiệu đầu ra:
- A_val[3:0] : giá trị số A đưa vào ALU.
- B_val[3:0] : giá trị số B đưa vào ALU.
- opcode[2:0] : mã lệnh đưa vào ALU (00 cộng, 01 trừ, 10 and, 11 or).
- en : tín hiệu enable bắt đầu tính toán cho ALU.

- bcd_opcode[2:0] : giá trị opcode đưa sang module hiển thị 7 đoạn.
- bcd_value[4:0] : giá trị số hiển thị trên LED (có thể là A, B hoặc kết quả).

Yêu cầu hoạt động:
- Controller dùng FSM 4 trạng thái:
- State 0: Nhập số A.
- State 1: Nhập số B.
- State 2: Nhập opcode.
- State 3: Kích hoạt ALU để tính toán, hiển thị kết quả.
- Người dùng nhấn nút SW để chuyển sang bước tiếp theo.
- Trong mỗi state nhập liệu, xoay rotary sẽ tăng/giảm giá trị tương ứng (A, B, opcode).
Ở state 3, Controller bật en để ALU chạy, khi ALU trả về ack thì kết quả được hiển thị ra LED.

Chu kỳ hiển thị:
- LED3 hiển thị opcode.
- LED2 và LED1 hiển thị giá trị (A hoặc kết quả ALU tùy theo state).

*/

module controller (
    input wire    clk_50Hz,  // clock lấy mẫu chậm từ rotary
    input wire    clk_40MHz, // clock hệ thống cho ALU
    input wire    rst_n,     // reset tích cực thấp

```

```
input wire    rotary_up, // xung xoay chiều lên
input wire    rotary_down, // xung xoay chiều xuống
input wire    sw_pressed, // xung nhấn nút (đã debounce)

output reg [3:0] A_val,    // số A
output reg [3:0] B_val,    // số B
output reg [2:0] opcode,    // opcode cho ALU
output reg     en,        // enable cho ALU
input wire     ack,        // ack từ ALU
input wire [3:0] alu_result, // kết quả từ ALU

output reg [2:0] bcd_opcode, // giá trị đưa sang hiển thị LED3
output reg [4:0] bcd_value    // giá trị đưa sang hiển thị LED2-LED1
);
```


