**Phân Tích Code i2c\_av\_config**

Code này là một **bộ điều khiển cấu hình I2C** (I2C Configuration Controller) cho một thiết bị ngoại vi, cụ thể là **Audio Codec SSM2603**.

🔹 **Mục đích**:

* **Gửi lệnh cấu hình qua I2C** để thiết lập các **thanh ghi điều khiển của Audio Codec**.
* **Gửi dữ liệu từ bảng tra cứu (LUT - Look-Up Table)** qua giao thức **I2C**.
* **Theo dõi trạng thái của quá trình cấu hình** (status).

**1. Phân Tích Các Thành Phần Chính**

**Khai báo input/output**

verilog

Sao chépChỉnh sửa

module i2c\_av\_config (

input clk, // Clock hệ thống

input reset, // Reset module

output i2c\_sclk, // Clock I2C

inout i2c\_sdat, // Dữ liệu I2C (2 chiều)

output [3:0] status // Theo dõi trạng thái cấu hình

);

* clk: **Clock chính** từ FPGA.
* reset: **Reset module**, đưa về trạng thái ban đầu.
* i2c\_sclk: **Clock I2C**, gửi tới Audio Codec.
* i2c\_sdat: **Dữ liệu I2C**, giao tiếp với Audio Codec.
* status: **Theo dõi trạng thái LUT**, cho biết đang cấu hình bước nào.

**Bảng LUT chứa dữ liệu cấu hình**

verilog

Sao chépChỉnh sửa

reg [23:0] i2c\_data; // Dữ liệu I2C để gửi

reg [15:0] lut\_data; // Dữ liệu từ LUT (Look-Up Table)

reg [3:0] lut\_index = 4'd0; // Chỉ số của LUT

parameter LAST\_INDEX = 4'ha;

* **lut\_data**: Chứa dữ liệu 16-bit sẽ gửi tới Audio Codec.
* **lut\_index**: Đếm số lệnh đã gửi (0 → LAST\_INDEX = 0xA).

**Dữ liệu LUT (lut\_data)**

verilog

Sao chépChỉnh sửa

always @(\*) begin

case (lut\_index)

4'h0: lut\_data <= 16'h0c10; // Bật tất cả ngoại trừ output

4'h1: lut\_data <= 16'h0017; // Cấu hình đầu vào trái

4'h2: lut\_data <= 16'h0217; // Cấu hình đầu vào phải

4'h3: lut\_data <= 16'h0479; // Cấu hình đầu ra trái

4'h4: lut\_data <= 16'h0679; // Cấu hình đầu ra phải

4'h5: lut\_data <= 16'h08d4; // Đường tín hiệu analog

4'h6: lut\_data <= 16'h0a04; // Đường tín hiệu số

4'h7: lut\_data <= 16'h0e01; // Giao diện âm thanh số

4'h8: lut\_data <= 16'h1020; // Tốc độ lấy mẫu

4'h9: lut\_data <= 16'h0c00; // Bật tất cả

4'ha: lut\_data <= 16'h1201; // Kích hoạt bộ xử lý

default: lut\_data <= 16'h0000;

endcase

end

📌 **Ý nghĩa**:

* Mỗi giá trị LUT là một **lệnh cấu hình I2C**, chứa **địa chỉ thanh ghi + dữ liệu**.
* **Mỗi lần gửi xong một lệnh, module sẽ gửi tiếp lệnh tiếp theo**.
* Giá trị LAST\_INDEX = 0xA là lệnh cuối cùng.

**Giao tiếp với bộ điều khiển I2C**

verilog

Sao chépChỉnh sửa

i2c\_controller control (

.clk (clk),

.i2c\_sclk (i2c\_sclk),

.i2c\_sdat (i2c\_sdat),

.i2c\_data (i2c\_data),

.start (i2c\_start),

.done (i2c\_done),

.ack (i2c\_ack)

);

* Module i2c\_controller chịu trách nhiệm **gửi dữ liệu qua I2C**.
* i2c\_data chứa dữ liệu 24-bit (8-bit địa chỉ + 16-bit dữ liệu từ LUT).
* i2c\_start: **Bắt đầu gửi dữ liệu I2C**.
* i2c\_done: **Báo hiệu hoàn tất truyền dữ liệu**.
* i2c\_ack: **Slave (Audio Codec) đã nhận dữ liệu**.

**Finite State Machine (FSM) Điều Khiển Cấu Hình**

verilog

Sao chépChỉnh sửa

reg [1:0] control\_state = 2'b00;

📌 **FSM có 4 trạng thái chính**:

* 00 → **Bắt đầu gửi lệnh**.
* 01 → **Chờ gửi xong**.
* 10 → **Kiểm tra ACK từ Codec**.
* 11 → **Kết thúc khi gửi hết LUT**.

**FSM gửi dữ liệu qua I2C**

verilog

Sao chépChỉnh sửa

always @(posedge clk) begin

if (reset) begin

lut\_index <= 4'd0;

i2c\_start <= 1'b0;

control\_state <= 2'b00;

end else begin

case (control\_state)

2'b00: begin

i2c\_start <= 1'b1;

i2c\_data <= {8'h34, lut\_data}; // Gửi địa chỉ 0x34 + dữ liệu LUT

control\_state <= 2'b01;

end

2'b01: begin

i2c\_start <= 1'b0;

control\_state <= 2'b10;

end

2'b10: if (i2c\_done) begin

if (i2c\_ack) begin

if (lut\_index == LAST\_INDEX)

control\_state <= 2'b11; // Hoàn tất

else begin

lut\_index <= lut\_index + 1'b1;

control\_state <= 2'b00; // Gửi tiếp lệnh kế tiếp

end

end else

control\_state <= 2'b00; // Gửi lại nếu không nhận được ACK

end

endcase

end

end

**2. Cách Hoạt Động**

1. **Bắt đầu (reset = 0)**
   * lut\_index = 0, control\_state = 00.
2. **Gửi lệnh (state = 00)**
   * i2c\_start = 1, gửi lệnh **đầu tiên** từ LUT ({8'h34, lut\_data}).
3. **Chờ hoàn thành (state = 01)**
   * Chờ i2c\_done = 1.
4. **Kiểm tra ACK (state = 10)**
   * Nếu **i2c\_ack = 1**, tăng lut\_index lên **gửi lệnh tiếp theo**.
   * Nếu **đã gửi hết (lut\_index = LAST\_INDEX)**, chuyển sang trạng thái 11 (hoàn tất).

📌 **Nếu không nhận được ACK (i2c\_ack = 0)**, thì gửi lại lệnh.

**3. Tổng Kết**

✅ **Chức năng chính**:  
🚀 **Tự động gửi lệnh cấu hình qua I2C để thiết lập Audio Codec**.  
🚀 **Sử dụng LUT để lưu danh sách lệnh cần gửi**.  
🚀 **FSM điều khiển từng bước của quá trình gửi dữ liệu**.

💡 **Ứng dụng thực tế**:

* Khi FPGA khởi động, module này tự động **cấu hình codec SSM2603** mà không cần CPU.
* Dùng trong **hệ thống xử lý âm thanh** như **DSP trên FPGA**.

🛠 **Bạn có thể kiểm tra hoạt động bằng mô phỏng (SignalTap hoặc Modelsim)!** 🚀