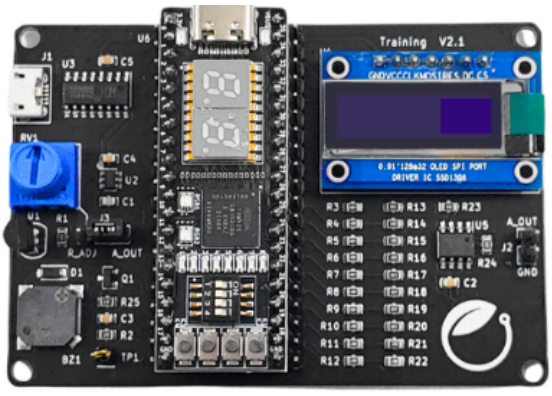
BÁO CÁO PROJECT
THIẾT KẾ SAR ADC TRÊN FPGA

1. Giới thiệu

Dự án này nhằm thiết kế và mô phỏng một bộ chuyển đổi tương tự – số (ADC) theo phương pháp **SAR (Successive Approximation Register)** với độ phân giải 8 bit, hoạt động trên nền tảng **FPGA**. Bộ ADC sử dụng tín hiệu so sánh từ bộ so sánh (Comparator) và thực hiện xấp xỉ nhị phân để tìm giá trị điện áp đầu vào.

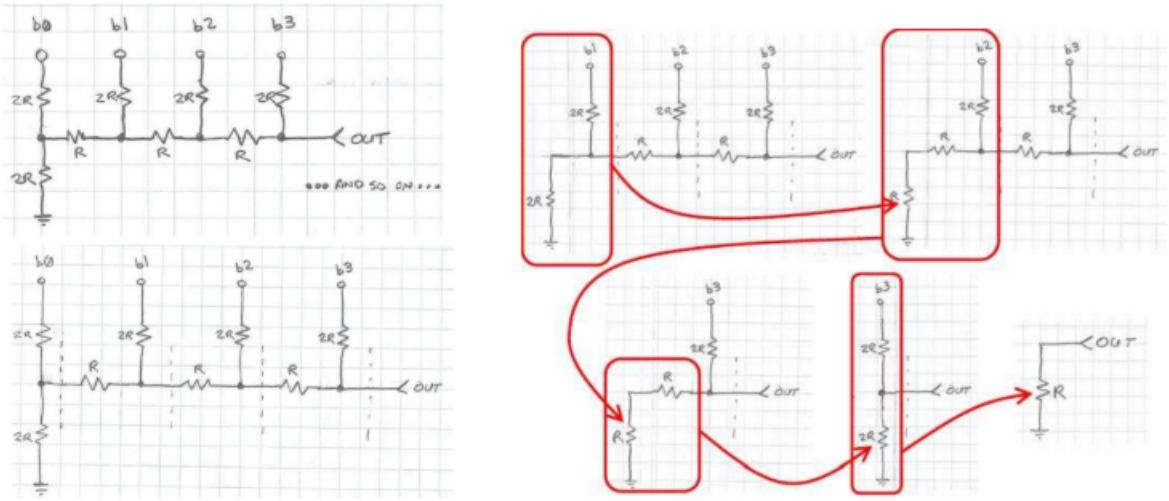
2. Thành phần hệ thống

- **Sample and Hold (S/H):** Lấy mẫu điện áp đầu vào tại một thời điểm cụ thể và giữ nguyên điện áp đó trong một khoảng thời gian.
- **SAR ADC (8-bit):** Thực hiện giải thuật xấp xỉ kế tiếp.
- **Clock Divider:** Chia tần số từ 12 MHz xuống tần số thấp để đồng bộ việc lấy mẫu.
- **Top Module:** Tích hợp SAR ADC và Clock Divider.
- **FPGA:** Sử dụng board FPGA STEP MAX10.

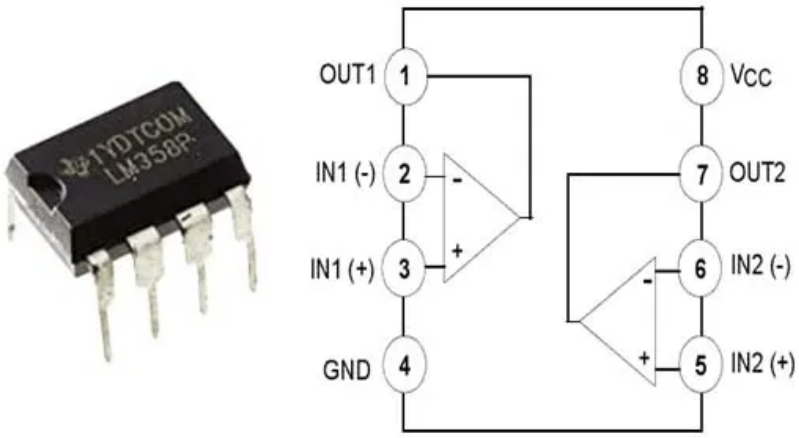


Hình 1: Board FPGA STEP MAX10

- **Comparator và DAC ngoài:** So sánh điện áp analog với tín hiệu DAC từ FPGA.

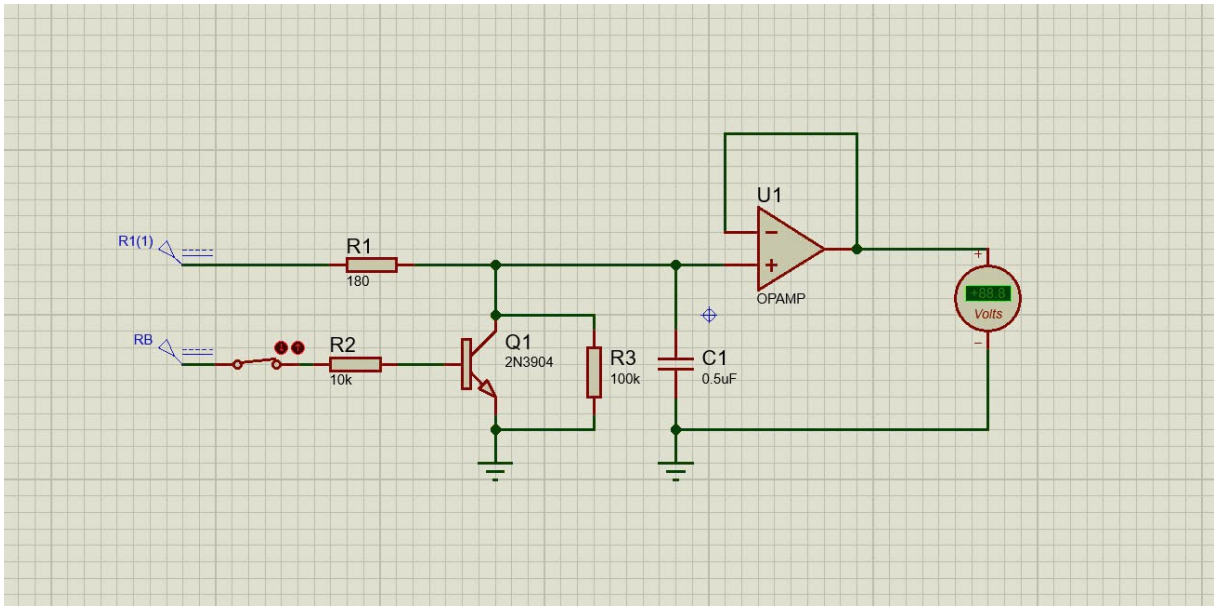


Hình 2: Bộ DAC dạng R-2R

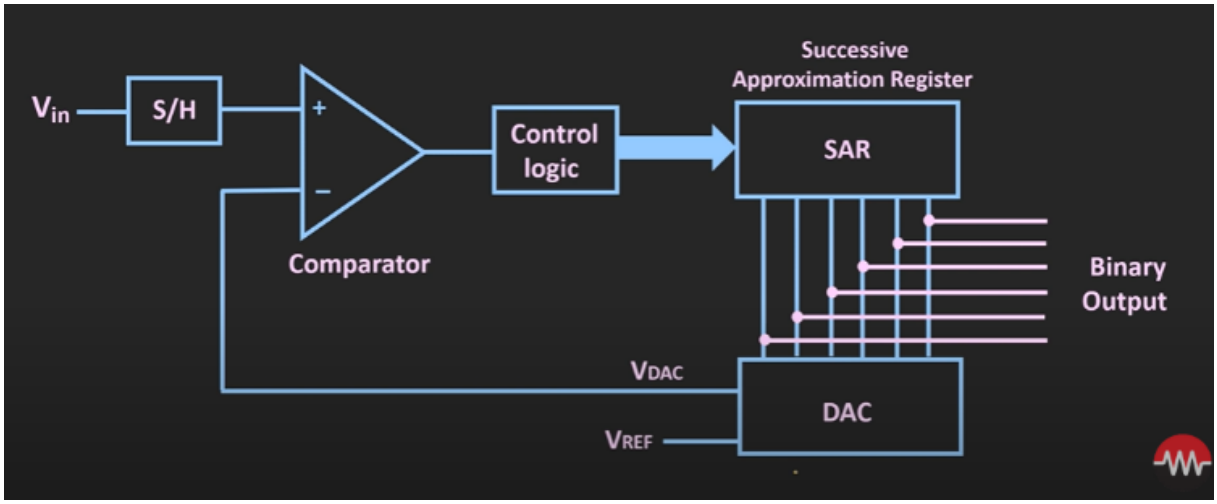


Hình 3: IC so sánh LM358P

3. Sơ đồ nối mạch (mô tả)



Hình 4: Mạch mô phỏng khối lấy và giữ mẫu



Hình 5: Sơ đồ tổng quan hệ thống SAR ADC

4. Mã nguồn (trích đoạn)

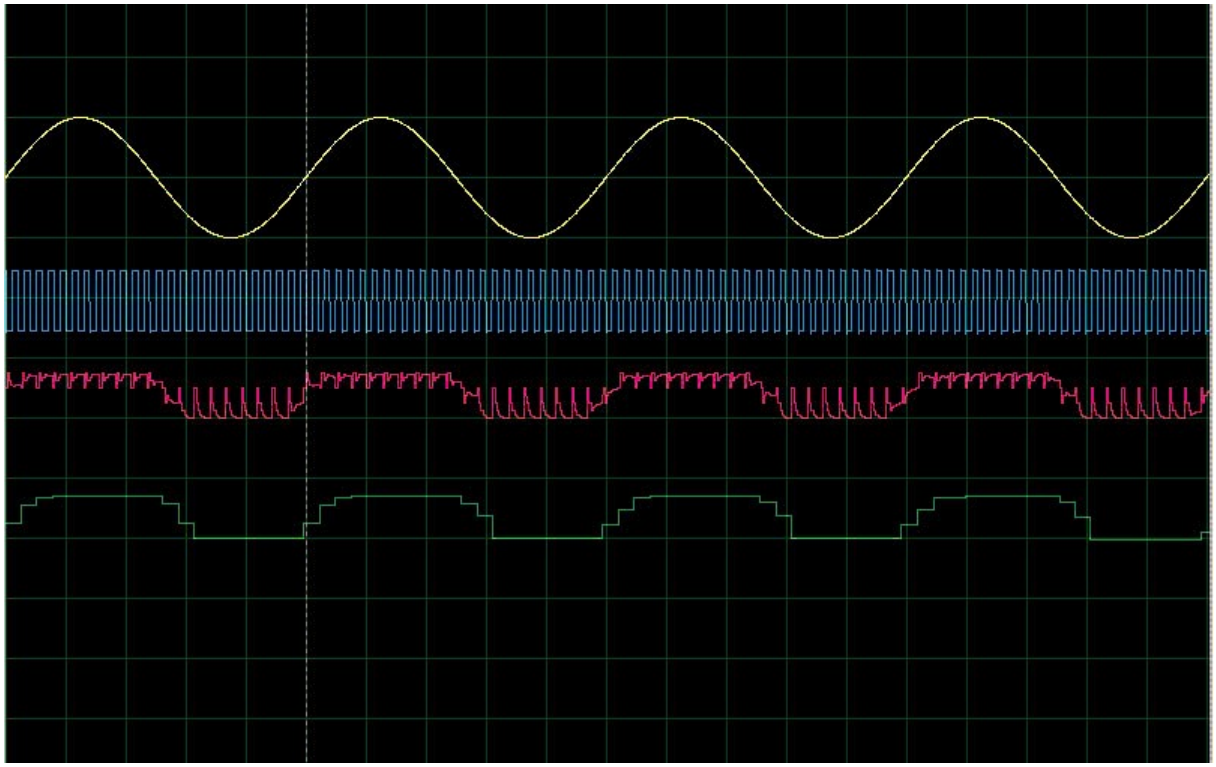
4.1 Mô-đun SAR ADC (Verilog)

```
// SAR ADC 8-bit with Clock Divider
module SAR_ADC #(parameter RESOLUTION = 8) (
    input wire clk_i,                   // Clock input (from clock divider
        )
    input wire rst_ni,                 // Asynchronous active-low reset
```

```
    input wire comp_i,                // Input from external analog
        comparator
    output wire rdy_o,                // Conversion complete signal
    output wire [RESOLUTION-1:0] dac_o // 8-bit digital output
);

    // State encoding
    localparam IDLE = 2'd0,
               CONVERT = 2'd1,
               DONE = 2'd2;

    // Internal registers
    reg [1:0] state_q, state_d;
    reg [RESOLUTION-1:0] mask_q, mask_d;
    reg [RESOLUTION-1:0] result_q, result_d;

    // Sequential logic
    always @(posedge clk_i or negedge rst_ni) begin
        if (!rst_ni) begin
            state_q <= IDLE;
            mask_q <= 1 << (RESOLUTION - 1);
            result_q <= 0;
        end else begin
            state_q <= state_d;
            mask_q <= mask_d;
            result_q <= result_d;
        end
    end

    // Combinational logic for SAR FSM
    always @(*) begin
        state_d = state_q;
        mask_d = mask_q;
        result_d = result_q;

        case (state_q)
            IDLE: begin
                state_d = CONVERT;
                mask_d = 1 << (RESOLUTION - 1);
                result_d = 0;
            end

            CONVERT: begin
                if (comp_i)
                    result_d = result_q | mask_q;
                else
                    result_d = result_q;

                mask_d = mask_q >> 1;
                state_d = (mask_d == 0) ? DONE : CONVERT;
            end

            DONE: begin
                state_d = IDLE;
            end

            default: begin
                state_d = IDLE;
```

```
            end
        endcase
    end

    assign dac_o = result_q;
    assign rdy_o = (state_q == DONE);

endmodule

// Clock Divider Module: divide 12 MHz input clock down to 4 Hz
module clock_divider (
    input wire clk,              // 12 MHz input clock
    output reg clk_4Hz = 0       // Output clock: 4 Hz
);

    // For simulation purposes, use smaller DIVISOR like 6_000 (for ~1
       kHz)
    parameter DIVISOR = 6_000;
    reg [21:0] counter = 0;

    always @(posedge clk) begin
        if (counter == DIVISOR - 1) begin
            counter <= 0;
            clk_4Hz <= ~clk_4Hz;
        end else begin
            counter <= counter + 1;
        end
    end
endmodule

// Top-level Module: integrates clock divider and SAR ADC
module ADC (
    input wire clk_12MHz,        // Main clock (12 MHz)
    input wire rst_ni,           // Asynchronous active-low reset
    input wire comp_i,           // Comparator input
    output wire rdy_o,           // ADC conversion ready
    output wire [7:0] dac_o      // Digital output (8-bit)
);

    wire clk_4Hz;

    // Instantiate clock divider
    clock_divider clk_div_inst (
        .clk(clk_12MHz),
        .clk_4Hz(clk_4Hz)
    );

    // Instantiate SAR ADC
    SAR_ADC #(8) adc_inst (
        .clk_i(clk_4Hz),
        .rst_ni(rst_ni),
        .comp_i(comp_i),
        .rdy_o(rdy_o),
        .dac_o(dac_o)
    );

endmodule
```

5. Kết quả mô phỏng



Hình 6: Kết quả mô phỏng hoạt động ADC