به نام خدا

نحوه ی اجرای کد ها:

در ابتدا ماژول AC_register را داریم که شامل ورودی های clk,rst,LD,INR,CLRو ورودی 16 بیتیDATE_INو همچنین یک خروجی 16بیتیAC_OUTکه بصورت REGتعریف شده است داریم.

در بخش دوم کد از ALWAYSاستفاده کرده و (*) به این معنی است که تغییرات روی همه ی ورودی ها اعمال شود .

شرط اول به این معنی است که اگر RSTزیاد باشد AC_OUT که یک 16 بیتی از نوع باینری است روی 0 تنظیم میشود.

اگر CLRزیاد باشد(RST) فعال نیست AC_OUTروی 0 تنظیم میشود.

اگر LDبالا باشدRST,CLRفعال نیستند AC_OUTبا DATA_INمقدار دهی میشود.

اگر INRبالا باشد و هیچ یک از شرایط قبلی فعال نباشد AC_OUTیک واحد افزایش میابد.

```
module AC_register (
    input clk,
    input rst,
    input LD,
    input INR,
    input CLR,
    input [15:0] data_in,
    output reg [15:0] AC_out
);
    always @(posedge clk or posedge rst) begin
        if (rst)
            AC_out <= 16'b0;
        else if (CLR)
            AC_out <= 16'b0;
        else if (LD)
            AC_out <= data_in;
        else if (INR)
            AC_out <= AC_out + 1;
    end
endmodule
```

غ

ماژول ADDER_LOGICALیک ورودی 16 بیتی به نام DRویک ورودی 8 بیتی به نام INPRدارد .ویک خروجی 16 بیتی به نام SUM_OUTدارد.

SUM_OUT خروجی + DRیک هشت بیتی از نوع باینری که مقدارش صفر است و با INPRجمع میشود .

ماژول CONTROL_GATES دارای ورودی های CLK,RST,LD,INR,CLR وخروجی 16 بیتی DR,AC_OUTو خروجی 8بیتی INPRاست.

WIRE:خروجی گیتی که ورودی گیت دیگری است.

یک خروجی WIREکه 16 بیت است به نام SUM_OUT

```
module adder_logical (
    input [15:0] DR,
    input [7:0] INPR,
    output [15:0] sum_out
);
    assign sum_out = DR + {8'b0, INPR};  // Assuming zero extension for INPR
endmodule

module control_gates (
    input clk,
    input rst,
    input LD,
    input INR,
    input CLR,
    input [15:0] DR,
    input [7:0] INPR,
    output [15:0] AC_out
);
    wire [15:0] sum_out;
```

در این بخش ماژول هارا فراخوانی کرده.

```
    // Instantiate adder_logical module
    adder_logical adder_log_inst (
        .DR(DR),
        .INPR(INPR),
        .sum_out(sum_out)
    );

    // Instantiate AC_register module
    AC_register AC_reg_inst (
        .clk(clk),
        .rst(rst),
        .LD(LD),
        .INR(INR),
        .CLR(CLR),
        .data_in(sum_out),
        .AC_out(AC_out)
    );
endmodule
```