

```

`define MSBI 13

module busqueda(
    input clk_fsm, //-----
    Reloj del video
    /*
        #python script
        window_size=40*10;
        window_limit=window_size;
        NumItera=0;
        numEst=10;
        time_per_frame=16e-3;
        while(window_limit>0):
            NumItera=NumItera+window_limit;
            window_limit=window_limit-1;
            NumItera=NumItera+window_size+1
            print(str(NumItera)+' clocks');
            clk_fsm=(1/(time_per_frame/(NumItera*numEst)))/1e6;
            print(str(clk_fsm)+' MHz using: ' +
str(1/time_per_frame)+ 'fps'

            ~@output shell:
                80601 clocks
                50.375625 MHz using: 62.5fps

        */
    input start, //-----
    Señal de inicio
    output finish, //-----
    Señal de finalizacion
    output idle,

    //frame de origen
    input [1:0]cont_img, //-----
    Contador imagen

    //HPS
    input vector_wait_fifo, //-----
    Señal de espera Fifo
    input img_wait_fifo, //-----
    Señal espera imagen fifo
    output[(`MSBI+2+`MSBI)+1:0]vector_me, //-----
    -----Salida vector con estimacion de movimiento
    output [25:0]img_mb, //-----
    imagen macrobloque
    output img_wr_req, //-----
    write request
    output vector_wr_req, //-----
    vector wirte request

    //RAM DE ACCESO
    input [24:0]data_rd_img_ref, //-----
    Datos de lectura imagen referencia

```

```

        input [24:0]data_rd_img_Act,/-------
Datos de lectura imagen actual
        output [`MSBI:0]add_read_img_ref,
        output [`MSBI:0]add_write_img_ref,
        output wr_enable_ref,//done
        output [`MSBI:0]add_read_img_act,
        output [`MSBI:0]add_write_img_act,
        output wr_enable_act,//done
        output [24:0]data_wr_img_ref,
        output [24:0]data_wr_img_Act,

        //limitador de ventana
        input [`MSBI:0]window_limit,

        output [4:0]real_state,
        output [`MSBI:0]_realact,
        output [`MSBI:0]_realref
    );

        //wr_ref__wr_act__img_wr_hps__vector_wr_hps__finish__idle__
        __incre_ref__incre_act__rst_ref__rst_act__state
localparam IDLE
=15'b0__0__0__0__0__0__1__0__
__0__1__1__00000;
localparam READ_MEM
=15'b0__0__0__0__0__0__0__0__
__0__0__0__00001;
localparam BUSCAR_SIMILAR
=15'b0__0__0__0__0__0__0__0__
__0__0__0__00010;
localparam GUARDAR_VECTOR_LOAD
=15'b0__0__0__0__0__0__0__0__
__0__0__0__00011;
localparam GUARDAR_VECTOR_WRITE
=15'b0__0__0__1__0__0__0__0__
__0__0__0__00100;
localparam SET_REF_BIT_AND_ACT_BIT_1_LOAD
=15'b0__0__0__0__0__0__0__0__
__0__0__0__00101;
localparam SET_REF_BIT_AND_ACT_BIT_1_WRITE
=15'b1__1__0__0__0__0__0__0__
__0__0__0__00110;
localparam INCREASE_REF_1
=15'b0__0__0__0__0__0__0__1__
__0__0__0__00111;
localparam INCREASE_REF_AND_ACT
=15'b0__0__0__0__0__0__0__1__
__1__0__0__01000;
localparam INCREASE_ACT
=15'b0__0__0__0__0__0__0__0__
__1__0__0__01001;
localparam SET_ACT2REF
=15'b0__0__0__0__0__0__0__0__
__0__0__0__01010;

```



```

assign real_state[4:0]=state[4:0];

assign _realact[`MSBI:0]=act[`MSBI:0];
assign _realref[`MSBI:0]=ref[`MSBI:0];

wire replace_act=state[14:0]==SET_ACT2REF;
always@(posedge clk_fsm, posedge rst_ref)
begin
    if(rst_ref)
    begin
        ref[`MSBI:0]<=0;
    end
    else
    begin
        ref[`MSBI:0]<=ref[`MSBI:0];
        if(incr_ref)
        begin
            ref[`MSBI:0]<=ref[`MSBI:0]+1'b1;
        end
    end
end

always@(posedge clk_fsm, posedge rst_act)
begin
    if(rst_act)
    begin
        act[`MSBI:0]<=0;
    end
    else
    begin
        act[`MSBI:0]<=act[`MSBI:0];
        if(incr_act)
        begin
            act[`MSBI:0]<=act[`MSBI:0]+1'b1;
        end
        else if(replace_act)
        begin
            act[`MSBI:0]<=ref[`MSBI:0];
        end
    end
end

always@(posedge clk_fsm)
begin
    case(state[14:0])
        IDLE :
        begin
            state[14:0]<=IDLE;
            if(start)
            begin
                state[14:0]<=READ_MEM;
            end
        end
        READ_MEM :

```

```

begin
    state[14:0]<=BUSCAR_SIMILAR;
    if(ref[`MSBI:0]>=window_limit[`MSBI:0])
    begin
        state[14:0]<=RESET_REF_BEFORE_SAVING_IMG;
    end
end
BUSCAR_SIMILAR :
begin
    if(data_rd_img_ref[7:0]!=data_rd_img_Act[7:0] )
    begin
        if(act[`MSBI:0]<window_limit[`MSBI:0])
        begin
            state[14:0]<=INCREASE_ACT;
        end
        else
        begin
            state[14:0]<=SET_REF_BIT_1_LOAD;
        end
    end
    else
    begin
        if(act[`MSBI:0]==ref[`MSBI:0])
        begin
            state[14:0]<=SET_REF_BIT_AND_ACT_BIT_1_LOAD;
        end
        else
        begin
            if(ref[`MSBI:0]>=window_limit[`MSBI:0])
            begin
state[14:0]<=RESET_REF_BEFORE_SAVING_IMG;
                end
            else
            begin
                state[14:0]<=GUARDAR_VECTOR_LOAD;
            end
        end
    end
end

end
GUARDAR_VECTOR_LOAD :
begin
    state[14:0]<=GUARDAR_VECTOR_WRITE;
    if(vector_wait_fifo)
    begin
        state[14:0]<=GUARDAR_VECTOR_LOAD;
    end
end

end
GUARDAR_VECTOR_WRITE :
begin
    state[14:0]<=SET_REF_BIT_AND_ACT_BIT_2_LOAD;

```

```

        if(vector_wait_fifo)
        begin
            state[14:0]<=GUARDAR_VECTOR_WRITE;
        end
    end
    SET_REF_BIT_AND_ACT_BIT_1_LOAD      :
    begin
        state[14:0]<=SET_REF_BIT_AND_ACT_BIT_1_WRITE;
    end
    SET_REF_BIT_AND_ACT_BIT_1_WRITE    :
    begin
        state[14:0]<=INCREASE_REF_AND_ACT;
    end
    INCREASE_REF_1                      :
    begin
        state[14:0]<=SET_ACT2REF;
    end
    INCREASE_REF_AND_ACT                :
    begin
        state[14:0]<=SET_ACT2REF;
    end
    INCREASE_ACT                        :
    begin
        state[14:0]<=READ_MEM;
    end
    SET_ACT2REF                        :
    begin
        state[14:0]<=READ_MEM;
    end
    SET_REF_BIT_AND_ACT_BIT_2_LOAD      :
    begin
        state[14:0]<=SET_REF_BIT_AND_ACT_BIT_2_WRITE;
    end
    SET_REF_BIT_AND_ACT_BIT_2_WRITE    :
    begin
        state[14:0]<=INCREASE_REF_1;
    end
    SET_REF_BIT_1_LOAD                  :
    begin
        state[14:0]<=SET_REF_BIT_1_WRITE;
    end
    SET_REF_BIT_1_WRITE                  :
    begin
        state[14:0]<=INCREASE_REF_1;
    end
    RESET_REF_BEFORE_SAVING_IMG         :
    begin
        state[14:0]<=LOAD_REF_2_IMG_PX;
    end
    LOAD_REF_2_IMG_PX                   :
    begin

        if(ref[`MSBI:0]>=window_limit[`MSBI:0])
        begin

```

```

        state[14:0]<=FINISH;
    end
    else
    begin
        if(img_wait_fifo)
        begin
            state[14:0]<=LOAD_REF_2_IMG_PX;
        end
        else
        begin
            state[14:0]<=WRITE_REF_2_IMG_PX;
        end
    end
end
WRITE_REF_2_IMG_PX
begin
    state[14:0]<=INCREASE_REF_2_IMG;
    if(img_wait_fifo)
    begin
        state[14:0]<=WRITE_REF_2_IMG_PX;
    end
end
INCREASE_REF_2_IMG
begin
    state[14:0]<=LOAD_REF_2_IMG_PX;
    if(ref[`MSBI:0]>=window_limit[`MSBI:0])
    begin
        state[14:0]<=FINISH;
    end
end
FINISH
begin
    state[14:0]<=IDLE;
end
default:
begin
    state[14:0]<=IDLE;
end
endcase
end

endmodule

```