Introduction to structured Vlsi

Teacher:Professor Lambert

Student:Peyman Pouyan

Objective:

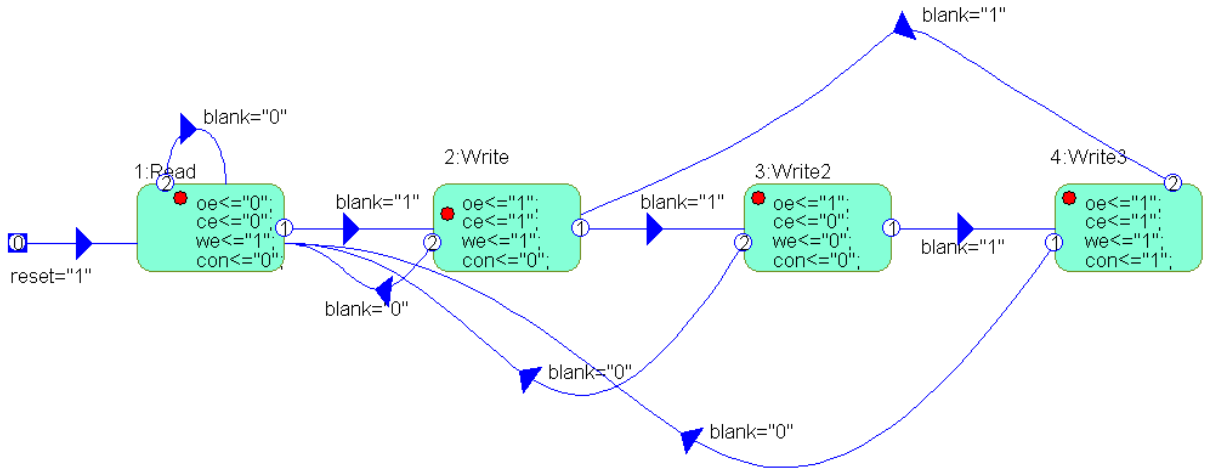
<u>Make it possible for a Cursor to move without leaving Traces in a coloured Background.</u>

Start:

We began this project by the first assignment which was making a coulerd screen with a rectangle in the middle of the screen.

What we learned from this assignment was a very simple counter and a simple state machine which controls writing and reading on and from fpga and also it controls the counter.

Lab 3 state machine:



Lab 3 counter:

```
architecture a0 of wgen is

begin

process(reset,cur_addr,con)
```

```
  begin
if reset="1" then
next_addr<=(others => '0');
elsif   con="1" then
next_addr<=cur_addr+1;
else
next_addr<=cur_addr;
            end if;       end process   ;    end a0;
```

Lab 4 and 5:

Here we needed to start with making a cursor and move it at first with leaving trace.following you can see the code for making the cursor and moving it with leaving trace.

```
architecture a0 of genwrite is
constant max_line : integer :=7;
constant max_pixel : integer :=2;
begin
process(reset,pixelingen,cnt_pixelingen,con_pgen)
            begin
                        if reset="1" then
                        pixeloutgen <= "000000";
                        cnt_pixeloutgen<= (others=>'0');
            elsif con_pgen="1" and cnt_pixelingen<max_pixel then
                        pixeloutgen<=pixelingen+1;
                        cnt_pixeloutgen<=cnt_pixelingen+1;
                        elsif          (cnt_pixelingen=max_pixel and con_pgen="1")  then
                                    pixeloutgen<="000000";
                                    cnt_pixeloutgen<= (others=>'0');
            else
                        pixeloutgen <= pixelingen ;
                        cnt_pixeloutgen<=cnt_pixelingen;
            end if;
```

```
end process;

process(reset,lineingen,cnt_lineingen,con_lgen)

 begin

            if reset="1"  then

                        lineoutgen<= "000000000";

                        cnt_lineoutgen<= (others=>'0');

            elsif  con_lgen="1"and cnt_lineingen<max_line  then

                        lineoutgen <= lineingen+1;

                        cnt_lineoutgen<=cnt_lineingen+1;

            elsif ( cnt_lineingen= max_line and con_lgen="1" ) then

                        lineoutgen <="000000000";

                        cnt_lineoutgen<=(others=>'0');

                       else

                        lineoutgen <= lineingen;

                        cnt_lineoutgen<=cnt_lineingen;

                        end if;

end process;


process(lineingen,pixelingen,next_line,next_pixel)

            begin

                        gen_writeaddress(5 downto 0)<=pixelingen(5 downto 0)+next_pixel(7 downto 2);

                        gen_writeaddress(14 downto 6)<=lineingen(8 downto 0)+next_line (8 downto 0);

end process;
```

Director:

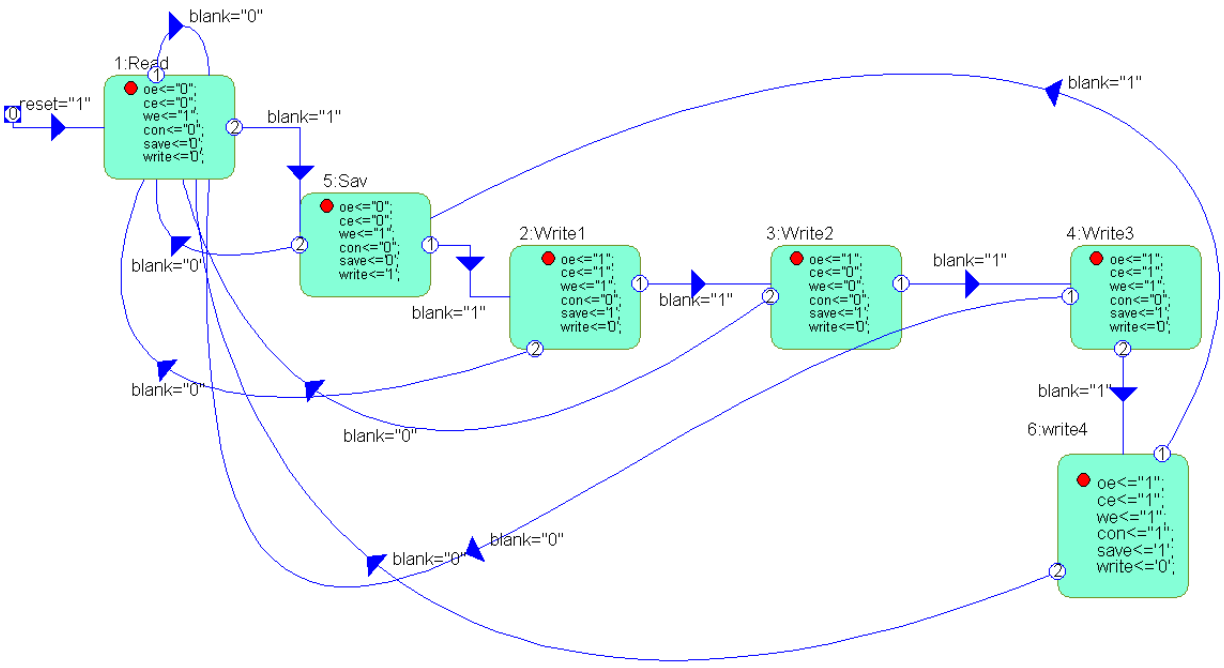
Left and right for example:

```
if LEFT="1"  then

                                    if curpixel = min_hcnt then

                                     nextpixel<= curpixel ;

                                     nextline<= curline;

                        else
```

```
                    nextline<= curline;
                    nextpixel <= curpixel - 1;
end if;
elsif RIGHT="1"  then
                        if curpixel = max_hcnt then
                nextpixel <= curpixel;
                        nextline <= curline;
                else
                        nextline <=curline;
                        nextpixel<= curpixel+ 1;
 end if;
```

We continued our design by trying to save the background in a place in the memory .With a lot of effort Finally we could do it.

The reason that it was difficult and time consuming to make this, was because we were using just one state machine which is difficult to deal with.



Steping in True Way:

By following professor lamberts advice we decided to change our design to Multi state controls.

So we can have a main state machine which controls other small state machines(Hierechical state machines)

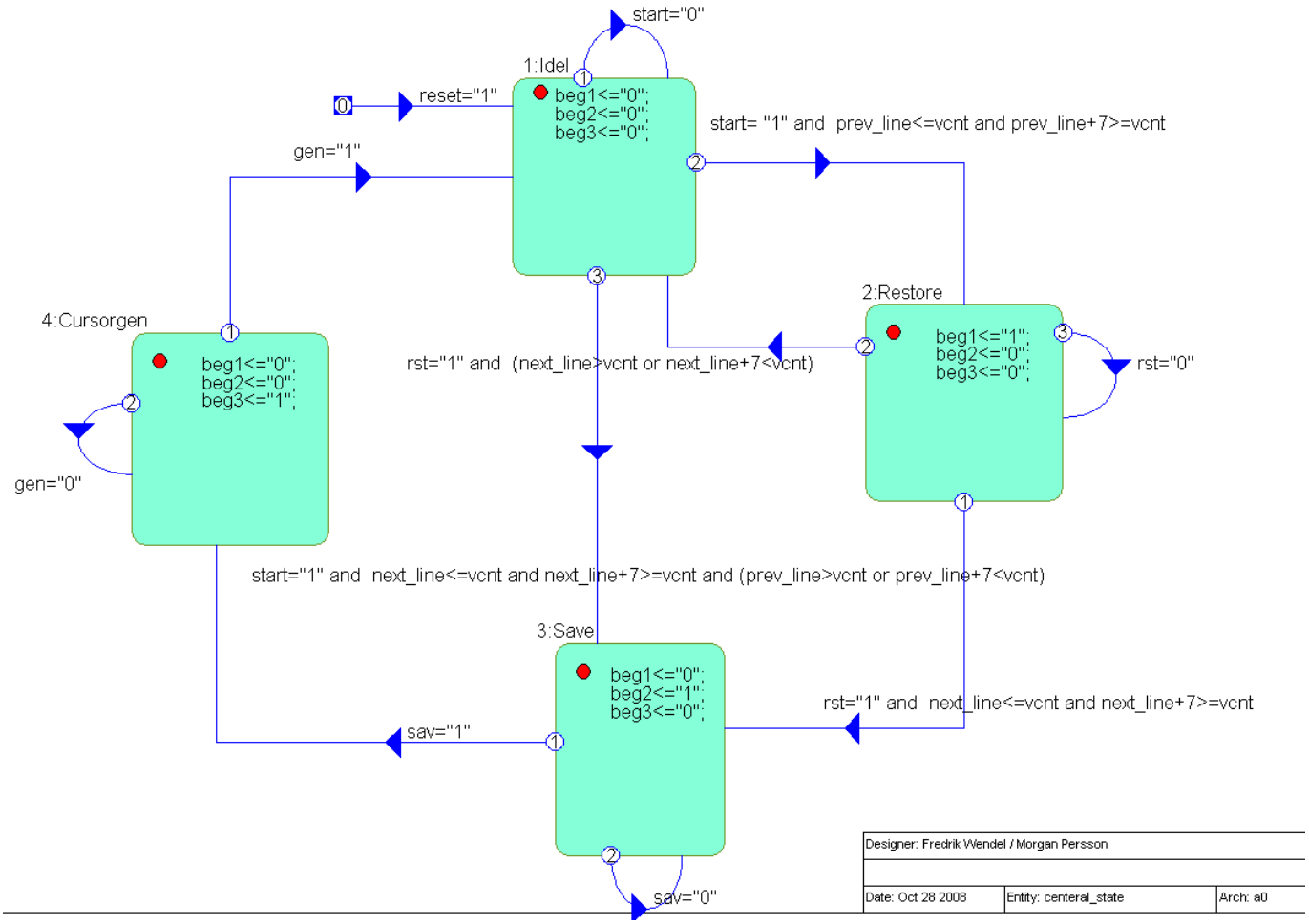
Power of eye in Image Diagnostic:

We know that Image is built from frames and when the speed of frames is fast our eye can not recognize the change in frames.So we decided to use this ability in order to have a simple and accurate design.So we use Vsync as an enable for change of contents,it means the place of cursor will only change with change of Vsync.

Main State machine:

The purpose is to generate the 8 lines of cursor in a frame and keeping it during the whole frame.So if in a frame we press the arrow key(up-down-right-left) it would not do anything and it would wait for the next frame to apply this movement.

Draw back is the design is easy and understandable but less fast.But as long as our eyes can not sense we did it.

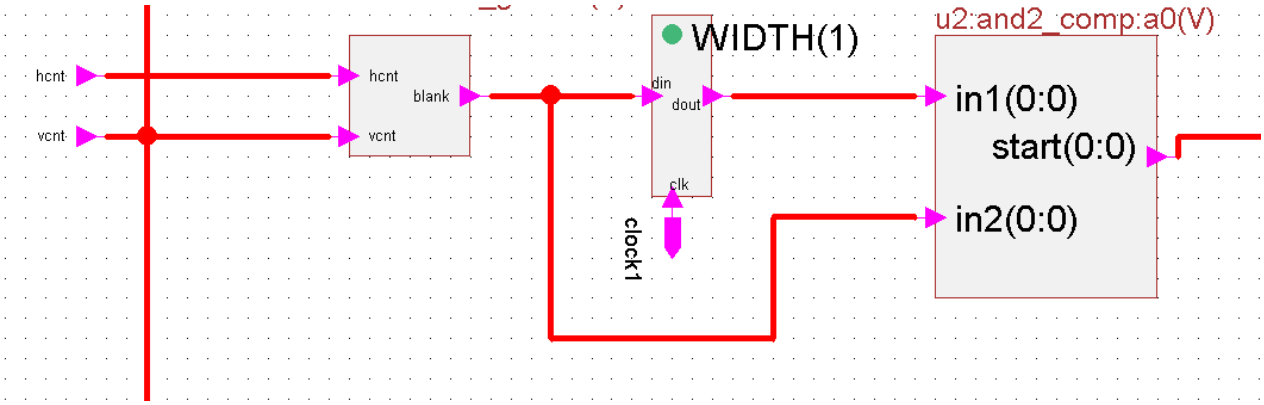


As you can see in the main state machine we have four parts :Idle -Restore-Save-Cursor generator

Each one of state machines(Restore-Save-Cursor generator) Fire another state machines which they control the whole process.

Some Digital Design techniques which we learned:

1-Making an edged detectin block:This permits us to find the time when blank become 1 and its previous value is zero.we called this signal zero.You can see blank edge detection below:



2-Ping-pong memory:Our algorithm had a problem when we press the up key ,so we used two places of memory which they exchange thier memory contents in a flash time.(We read from one and write from the other one and they exchange their content)

3-Using "**Or"** instead of making multyplexer which helps the design be more easier and consumes less area.

Final Assignment:

With this design it was so easy to do the final assignment .What needed was to make background and make the cursor move pixelwise.

Pixel wise movement:

It is just making the cursor to move in a area of 3 bytes (which we save).

So by looking at the byte of address in a horizontal line I make the suitable cursor area and back ground area.

Then by just puting this suitable virtual cursor and virtual background in a good order it was possible to move the cursor pixel wise

Curor Data=(Real cursor **and** virtual background) **or** virtual cursor.

Following you can see generation of virtual background(v) and virtual cursor(c).

0	0	C	C	c	c	c	c	c	c	c	c	c	c	C	c	c	c	0	0	0	0	0	0

0	0	0	0	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	c	0	0	0	0

0	0	C	c	c	c	c	c	c	c	c	c	c	c	c	c	c	0	0	0	0	0	0
1	1	V	v	v	v	v	v	v	v	v	v	v	v	v	v	v	1	1	1	1	1	1
1	1	1	1	v	v	v	v	v	v	v	v	v	v	v	v	v	v	v	1	1	1	1
1	1	1	1	1	1	v	v	v	v	v	v	v	v	v	v	v	v	v	v	1	1	1	1

Background:

The generation of background is simply a counter which generates the whole addresses of screen and it starts when blank(for me start) becomes "1" and when the background is finished I start the task of restore –save and generation(as my main state controls before).

Results:

Xilinx Frequency: 13.346Mhz

Number of CLBS: 86%

Number of IOBS: 70%

Delay of pressing arrow key and last write:Testpin 1: 14.10ms

Delay in modelsim:12.20ms