**Technion**

*Electrical Engineering Department*

High Speed Digital Systems Lab

**Menu Navigation Project**

**Characterization Documentation**

**Student: Tzahi Ezra**

**Advisors: Netanel Yamin**

**Moshe Porian**

**Last updated: 31.05.2014**

**Characterization document:**

1. General scheme

The following scheme includes the navigator block added to the SG TOP block,

y\_software

x\_software

debouncer

debouncer

debouncer

debouncer

software synchronization

right

x\_y\_ location\_top

To manager

RIGHT

left

hor\_out

LEFT

UP

To manager

ver\_out

up

DOWN

down

From navigator block

sym\_col

navigator block

From navigator block

ver\_out

hor\_out

sdram\_mux\_out[7:0]



sym\_row

ram\_data\_out[12:0]

Cond

Counters

FSM

fifo\_b\_wr\_en

fifo\_b\_rd\_en

sdram\_rd\_en\_out

sdram\_addr\_rd[23:0]

Fifo\_b\_data\_in[7:0]

ram\_addr\_rd[8:0]

fifo\_a\_data\_in[7:0]

fifo\_a\_wr\_en

fifo\_a\_rd\_en

ram\_rd\_en\_out

Manager

Symbol inversion

clk

reset

req\_in\_trg

mng\_en

sdram\_data[7:0]

sdram\_data\_valid

fifo\_a\_empty

fifo\_b\_empty

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Description** | **Width (bits)** | **Signal Type** | **Direction** | **Signal name** |
| System clock (100MHz) | 1 | std\_logic | in | clk |
| Asynchronous reset. Active low. | 1 | std\_logic | in | reset\_n |
| Count right Enable | 1 | std\_logic | in | right |
| Count left Enable | 1 | std\_logic | in | left |
| Count up Enable | 1 | std\_logic | in | up |
| Count down Enable | 1 | std\_logic | in | down |
| Horizontal location output | hor\_width\_g | std\_logic\_vector( hor\_width\_g-1 downto 0) | out | hor\_out |
| Vertical location output | ver\_width\_g | std\_logic\_vector( ver\_width\_g-1 downto 0) | out | ver\_out |
| Horizontal location Synchronized value with the software | hor\_width\_g | std\_logic\_vector( hor\_width\_g-1 downto 0) | out | x\_software |
| Vertical location Synchronized value with the software | ver\_width\_g | std\_logic\_vector( ver\_width\_g-1 downto 0) | out | y\_software |

|  |  |  |  |
| --- | --- | --- | --- |
| **Generic parameter** | | |  |
| **Name** | **Type** | **description** | **Default value** |
| hor\_width\_g | positive | Defines the horizontal width of Hor\_out lines needed to hold the maximum value kept in the counter | 5 |
| ver\_width\_g | positive | Defines the vertical width of Ver\_out lines needed to hold the maximum value kept in the counter | 4 |

1. symbol\_inversion:

Marking of a symbol would be responded by inverting the symbol's pixels (32\*32).

cond

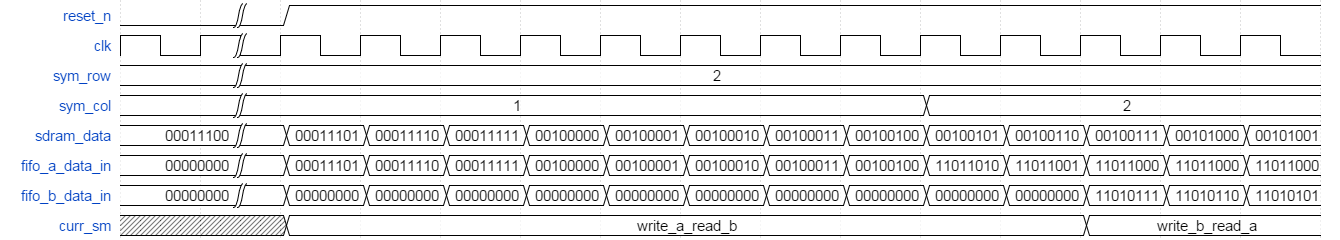
pixel (7..0)

data\_out(7..0)

Symbol inversion

In the manager block, the information of each pixel of the chosen symbol, being read from the SDRAM is inverted before being written to the fifos and passed to the VESA display. Using a synchronous mux as shown in the figure above. The Cond would be '1' if the internal row and column that are being calculated (represent the displayed symbol) matches the desired cursor symbol's location, which would be discussed later on.

Following presented a simulation showing the inversion operation correctness, in the manager block, when the manager FSM, responsible for toggling between fifos a and b, is in state write\_a\_read\_b, or write\_b\_read\_a, and when the internal row and column count are both 2, the corresponding outputs fifo\_a\_data\_in and fifo\_b\_data\_in, receives the inverted pixel input from the sdram, sdram\_data:



1. Debouncer

**Functionality:**

The debouncer is a counter which counts how long its input is '1', and triggers when it reaches a defined value (in our case, 0.5 sec is used). It works with the system frequency (100 MHz) such that when it reaches the defined value (50000000 clk cycles), its output turns from '0' to '1' for 1 period, than return to '0'.

**Block Top Diagram:**

clk

debouncer

reset

dout

din

|  |  |  |  |
| --- | --- | --- | --- |
| **Component generics** | | |  |
| **Name** | **Type** | **description** | **Default value** |
| max\_value\_g | positive | Number of clk cycles to wait, before output a rectangular Pulse in width of one cycle. | 50000000 |
| reset\_polarity\_g | std\_logic | The reset polarity of the system | '1' |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Description** | **Width (bits)** | **Signal Type** | **Direction** | **Signal name** |
| system clock (100MHz) | 1 | std\_logic | in | clk |
| Asynchronous reset | 1 | std\_logic | in | reset |
| Input from a button on the DE2 board | 1 | Std logic | in | din |
| Debouncer data output | 1 | std logic | out | dout |

1. x\_y\_location\_top

**Functionality:**

This block consists of 2 building blocks: x\_y\_location block and update upon vsync block. The first is used to determine and hold the cursor location due to buttons presses, and the second is used to update the location, only upon vsync signal arrival.

**Block Top Diagram:**

x\_y\_

location\_top

hor\_width\_g

clk

x\_updated

up\_trig

down\_trig

reset

ver\_width\_g

right\_trig

left\_trig

y\_updated

vsync

**Block implantation:**

down\_trig

right\_trig

x\_y\_location\_top

clk

ver\_width\_g

hor\_width\_g

x\_updated

reset

up\_trig

left\_trig

y\_updated

vsync

hor\_width\_g

x\_out

update\_ upon\_ vsync

x\_y\_

location

y\_out

ver\_width\_g

|  |  |  |  |
| --- | --- | --- | --- |
| **Component generics** | | |  |
| **Name** | **Type** | **description** | **Default value** |
| hor\_width\_g | positive | Defines the width of X\_out lines needed to hold the maximum horizontal location value | 5 |
| ver\_width\_g | positive | Defines the width of Y\_out lines needed to hold the maximum value kept in the counter | 4 |
| reset\_polarity\_g | std\_logic | The reset polarity of the system | '1' |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Description** | **Width (bits)** | **Signal Type** | **Direction** | **Signal name** |
| System's clock (100MHz) | 1 | std\_logic | in | clk |
| The system's reset | 1 | std\_logic | in | reset |
| Vsync Signal from the vesa gen ctrl block, indicates start of frame display, from the upper left corner on screen | 1 | std\_logic | in | vsync |
| Right Enable | 1 | std\_logic | in | right\_trig |
| Left Enable | 1 | std\_logic | in | left\_trig |
| Up Enable | 1 | std\_logic | in | up\_trig |
| Down Enable | 1 | std\_logic | in | down\_trig |
| The updated upon vsync horizontal location of the cursor in the frame. | hor\_width\_g | std\_logic\_vector(hor\_width\_g -1 downto 0) | out | x\_updated |
| The updated upon vsync vertical location of the cursor in the frame. | ver\_width\_g | std\_logic\_vector(ver\_width\_g -1 downto 0) | out | y\_updated |

1. x\_y\_location

**Functionality:**

The location of the cursor symbol is held by this block. It works with a clock in the system frequency (100 MHz), and it changes the location of the cursor symbol according to a state table, listed below.

**Block top diagram:**

hor\_width\_g

clk

x\_out

up\_trig

down\_trig

right\_trig

y\_out

ver\_width\_g

reset

x\_y\_

location

left\_trig

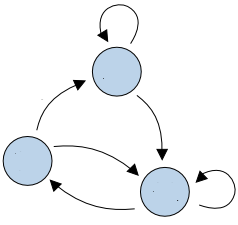
**Block implantation:**

The block uses an FSM to manage its operation.

The following diagram represent the block's inner implementation:

clk

right\_trig

FSM

x\_out

reset

left\_trig

X

x\_y\_location

Y

y\_out

up\_trig

down\_trig

**x\_y\_location states table:**

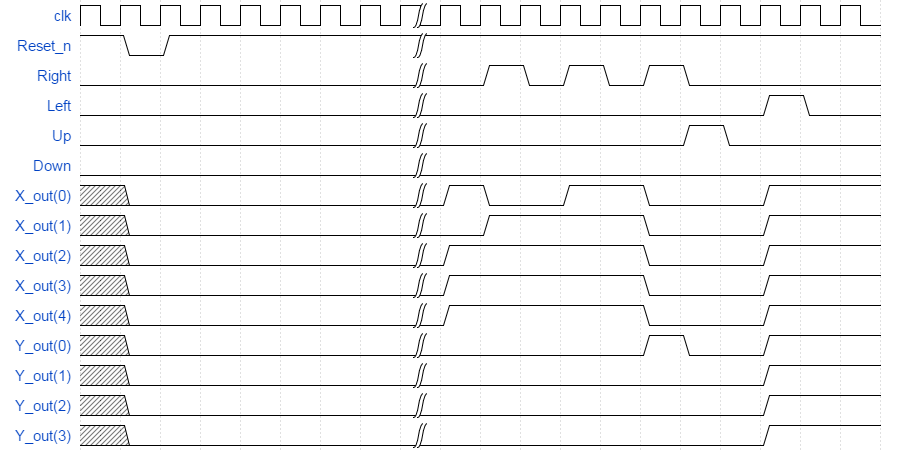
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **down** | | | **up** | | **left** | | | **right** | | |  | | |
| **Next state** | | | | | | | | | | | **Curr. state** | | |
| **Y** | **X** | **Y** | | **X** | | **Y** | **X** | | **Y** | **X** | | **Y** | **X** |
| **1** | **0** | **14** | | **1** | | **14** | **19** | | **0** | **1** | | **0** | **0** |
| **1** | **19** | **14** | | **0** | | **0** | **18** | | **1** | **0** | | **0** | **19** |
| **0** | **18** | **13** | | **19** | | **14** | **18** | | **0** | **0** | | **14** | **19** |
| **0** | **19** | **13** | | **0** | | **13** | **19** | | **14** | **1** | | **14** | **0** |
| **1** | **X** | **14** | | **X+1** | | **0** | **X-1** | | **0** | **X+1** | | **0** | **1..18** |
| **19** | **Y+1** | **19** | | **Y-1** | | **Y-1** | **18** | | **Y+1** | **0** | | **1..13** | **19** |
| **0** | **X-1** | **13** | | **X** | | **14** | **X-1** | | **14** | **X+1** | | **14** | **1..18** |
| **Y+1** | **0** | **Y-1** | | **0** | | **Y-1** | **19** | | **Y** | **1** | | **1..13** | **0** |
| **Y+1** | **X** | **Y-1** | | **X** | | **Y** | **X-1** | | **Y** | **X+1** | | **1..13** | **1..18** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Component generics** | | |  |
| **Name** | **Type** | **description** | **Default value** |
| hor\_width\_g | positive | Defines the width of X\_out lines needed to hold the maximum horizontal location value | 5 |
| ver\_width\_g | positive | Defines the width of Y\_out lines needed to hold the maximum value kept in the counter | 4 |
| reset\_polarity\_g | std\_logic | The reset polarity of the system | '1' |
| hor\_max\_value | positive | The maximum horizontal location value. | 19 |
| ver\_max\_value | positive | The maximum vertical location value. | 14 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Description** | **Width (bits)** | **Signal Type** | **Direction** | **Signal name** |
| System's clock (100MHz) | 1 | std\_logic | in | clk |
| The system's reset | 1 | std\_logic | in | reset |
| Right Enable | 1 | std\_logic | in | right\_trig |
| Left Enable | 1 | std\_logic | in | left\_trig |
| Up Enable | 1 | std\_logic | in | up\_trig |
| Down Enable | 1 | std\_logic | in | down\_trig |
| The future horizontal location of the cursor in the frame | hor\_width\_g | std\_logic\_vector(hor\_width\_g -1 downto 0) | out | x\_out |
| The future vertical location of the cursor in the frame | ver\_width\_g | std\_logic\_vector(ver\_width\_g -1 downto 0) | out | y\_out |

**x\_y\_location operation example:**

The following diagram shows the block's wave form. Special cases where reset\_n is low, outputs turn to be all zeros. When Right is triggered while x\_out="11111" and y\_out="0000", they turn to: x\_out="00000" and y\_out="0001", when Left is triggered while x\_out="00000" and y\_out="0000", they turn to: x\_out="11111" and y\_out="11111".



1. **update\_apon\_vsync**

**Functionality:**

This block responsible for outputting an updated X, Y values for cursor location, only upon vsync signal arrival.

When vsync signal is triggered high, the horizontal and vertical location output would take x, y values accordingly, when reset is triggered high x, y would be all zeros, and else the outputs keep their previous values.

**Block Top Diagram:**

reset

update\_ upon\_ vsync

x\_updated

hor\_width\_g

vsync

x

y

y\_updated

ver\_width\_g

|  |  |  |  |
| --- | --- | --- | --- |
| **Component generics** | | |  |
| **Name** | **Type** | **description** | **Default value** |
| hor\_width\_g | positive | Defines the width of X\_out lines needed to hold the maximum horizontal location value | 5 |
| ver\_width\_g | positive | Defines the width of Y\_out lines needed to hold the maximum value kept in the counter | 4 |
| reset\_polarity\_g | std\_logic | The reset polarity of the system | '1' |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Description** | **Width (bits)** | **Signal Type** | **Direction** | **Signal name** |
| The system's reset | 1 | std\_logic | in | reset |
| Vsync Signal from the vesa gen ctrl block, indicates start of frame display, from the upper left corner on screen | 1 | std\_logic | in | vsync |
| The future horizontal location of the cursor in the frame. | hor\_width\_g | std\_logic\_vector(hor\_width\_g -1 downto 0) | in | x |
| The future vertical location of the cursor in the frame. | ver\_width\_g | std\_logic\_vector(ver\_width\_g -1 downto 0) | in | y |
| The updated upon vsync horizontal location of the cursor in the frame. | hor\_width\_g | std\_logic\_vector(hor\_width\_g -1 downto 0) | out | x\_updated |
| The updated upon vsync vertical location of the cursor in the frame. | ver\_width\_g | std\_logic\_vector(ver\_width\_g -1 downto 0) | out | y\_ updated |