In order to make sure that our robot is secure for a real environment, we should test the more dangerous modules. One of it can be the PID module.

The steps that we will follow are represented below:

1)We build a VGA module to see the expected results like a first aproximation.

2)We will use two potentiometer like an input. We will obtain the product of its values and show it in the screen (Using for that the above module)

3) Now, it is the moment to configure the PID control

Now we are going to explain each step separately:

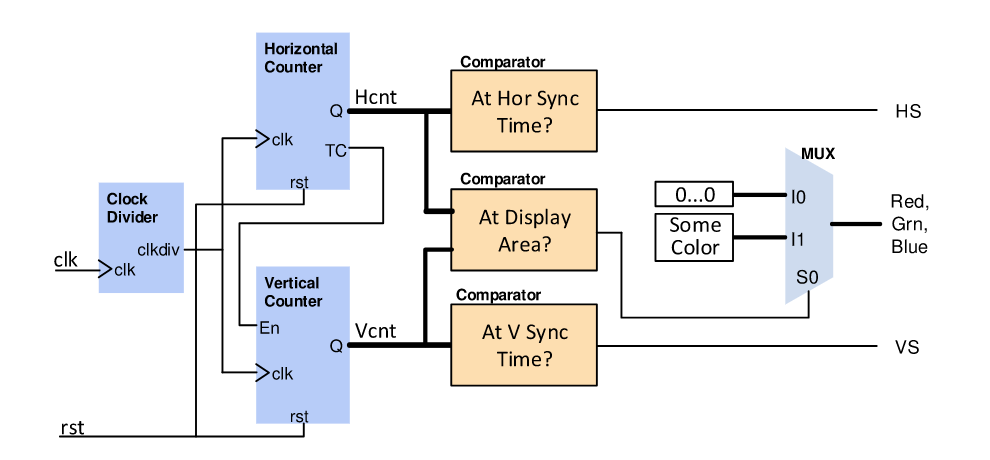
1. VGA Module

You can visit the following link to get a first basic module. This module shows an unique color in the screen but we want to achieve a representation of a number.

In order to do so, it is necessary to know how it work a VGA controller.

It exists three signals, red, green and blue wich send information to a VGA monitor. The totality of the colors can be obtain with a lineal comination of them.

A pixel do not have a lot of information to make an image, a horizontal line has a little more and with a matrix of pixels we can begin building an image. A VGA matrix has 480 rows with a total of 640 columns (pixels). So as to set an image in the screen it is required two synchronization signals, one for each axis.



We will build three independents module. A summary will be:

-hvsync\_generator: This module is responsable for generating the two synchronization signals for the vertival and horizontal axis.

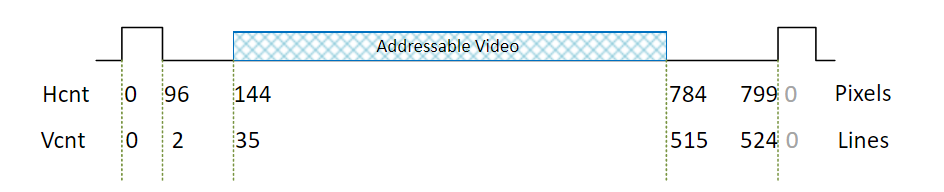
-RGB: This module contains the necessary instructions in order to access to the next block and to deploy the text in pixels matrix.

-Font: This module contains the fonts that we will use.

Now, we study each module separately.

**HVSYNC\_GENERATOR MODULE**

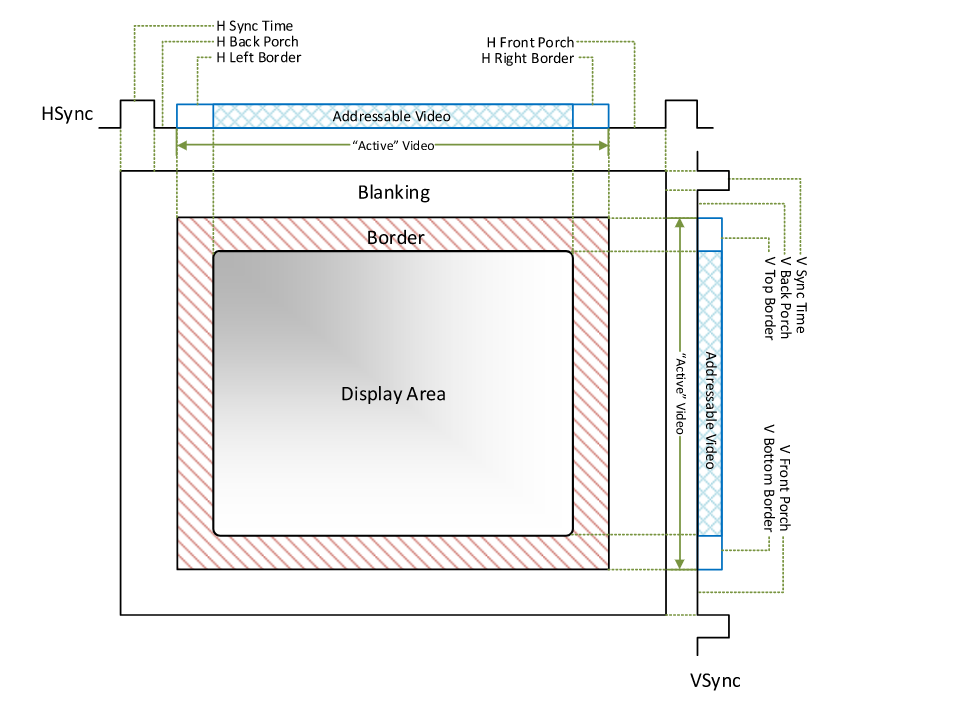
We need two counters, a counter (horizontal counter) to count pixels in each line and another counter (vertical counter) to count lines in a frame. The horizontal counter needs to reset itself when it reaches the end of the line (799 in this case). Similarly, vertical counter needs to reset itself when it reaches the end of the frame. The following image shows the HS and VS generation based on the counter values.



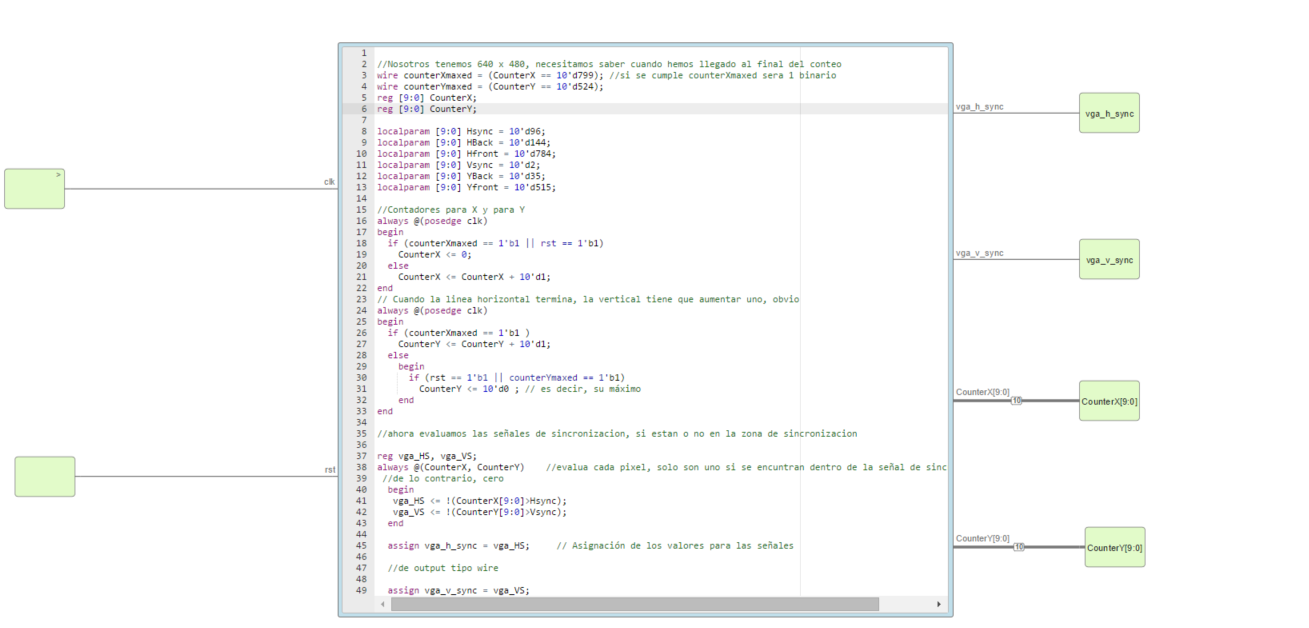
Input: clock and reset.

Output: H\_sync, V\_sync, CounterX and CounterY

Behaviour: If we are evaluating the horizontal pixels between 0 and 96, HSync will be 1, in other case, will be 0. If we are evaluating the vertical pixels between 0 and 2, VSync will be 1, in other case, will be 0. We get two synchronization signals.



This will be our first module:



In order to test it we have acquired a Hantek osciloscopie like de following:

**RGB MODULE**

Input: R,G,B signals. In ordert to test this part, like an first aproximation, RGB will be defined with the switches in IceZum Alhambra. It means, eight posible colors. Also, we will have CounterX and CounterY and a new variable called rom\_bit wich will have the pixel that we need to get the number.

Output: R,G,B signals for VGA interfaz, rom\_addr, rom\_col and character\_select.

Enviorment: output are the input in the next module in order to calculate rom\_bit.

**INPUT MODULE**

https://github.com/Obijuan/MonsterLED/wiki

<https://olimex.wordpress.com/2016/07/19/fpga-tutorial-vga-video-generation-with-fpga-and-verilog-add-video-memory-to-the-project-and-object-animation/>

<http://www2.elo.utfsm.cl/~lsb/elo211/labs/2006/elo212-lab08-0106.pdf>

<https://electr0nicablog.wordpress.com/2016/09/01/controladorvga/>

<http://martin.hinner.info/vga/timing.html>

<https://learn.digilentinc.com/Documents/269>

<http://www.epanorama.net/faq/vga2rgb/calc.html>

<https://www.monitortests.com/pixelclock.php?width=540&height=380&refresh=72&rate=1&decimals=2&minhblank=56&maxhblank=480&hmultiple=8&minvblank=3&maxvblank=120&vmultiple=1&maxpclock=165>