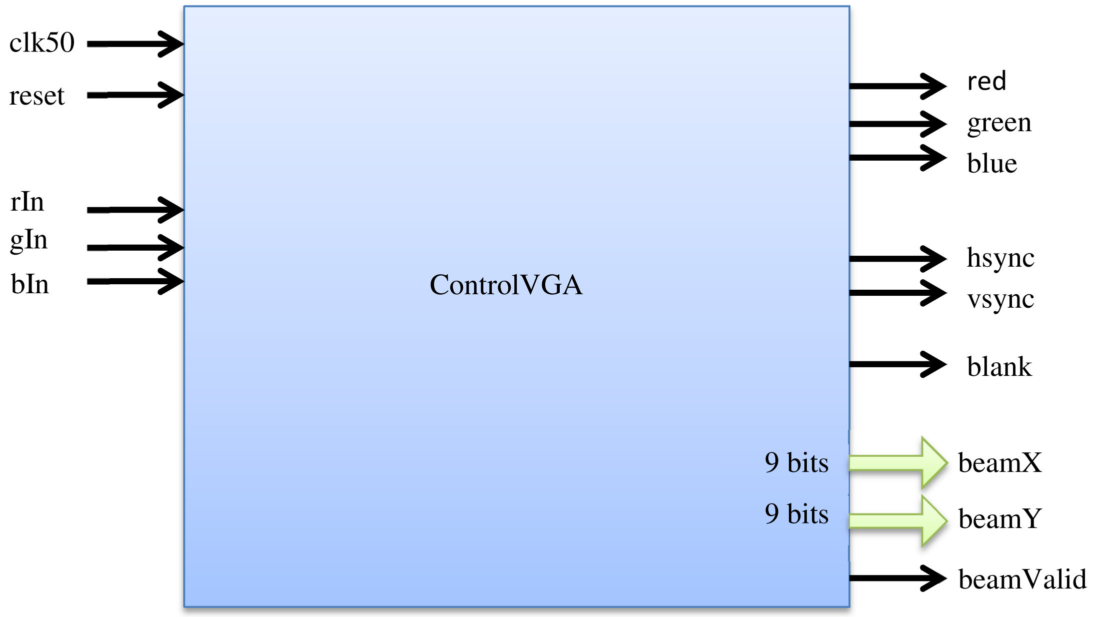
# TP VGA COUNTER

## The structural view :

The goal here, is too display a digit on a VGA screen (your computer screen). To reach this goal, we provide you 2 VHDL boxes.

## The VGA controller

The first one is the VGA controller. This box drive signals of the VGA interface to display pixel on the VGA screen.



**Input signals:**

| | |
|---|---|
| clk50 | 50 MHz clock |
| reset | reset the system when  high (‘1’) |
| rIn | Red In. This signal must be high when you want to light the red pixel at coordinate (beamX, beamY) |
| gIn | Green In. This signal must be high when you want to light the green pixel at coordinate (beamX, beamY). |
| bIn | Green In. This signal must be high when you want to light the blue pixel at coordinate (beamX, beamY). |

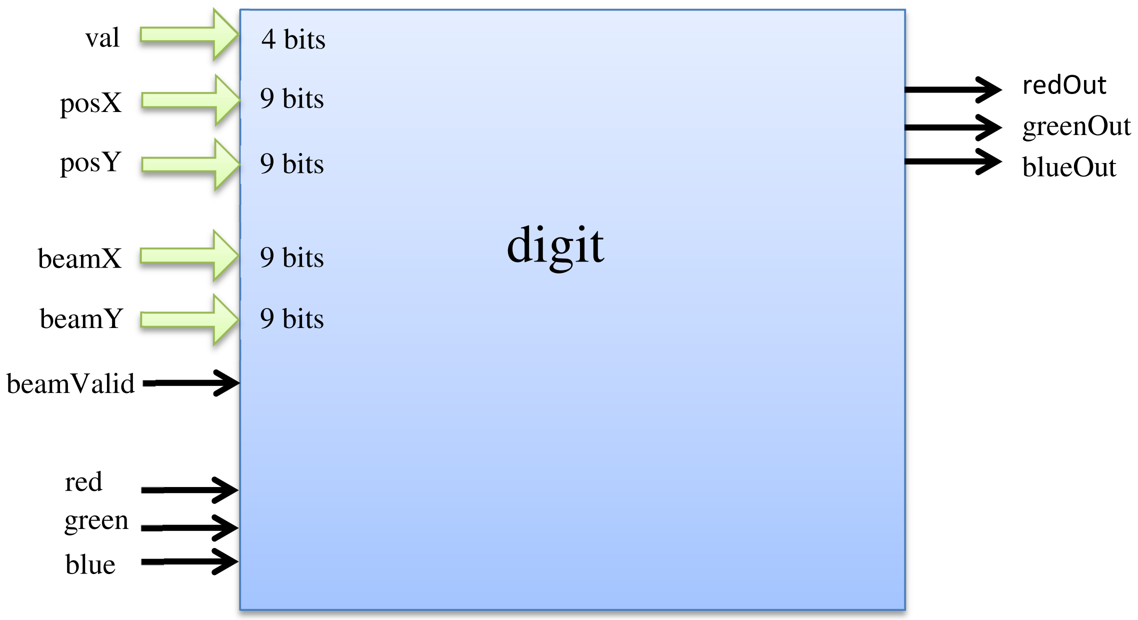
**Ouput signals:**

| | |
|---|---|
| Red | Red signal of the beam (refreshed pixel) |
| Green | Green signal of the beam (refreshed pixel) |
| Blue | Blue signal of the beam (refreshed pixel) |

| | |
|---|---|
| Hsync | Horizontal synchronization signal |
| Vsync | Vertical synchronization signal |
| blank | Means the beam is outside the visible area (640x480) |
| beamX | X coordinate of the beam (refreshed pixel) within the visible display (0 to 640) |
| beamY | Y coordinate of the beam (refreshed pixel) within the visible display (0 to 480) |
| beamValid | When high (‘1’) : the beam is in the visible area (640x480 area) |

## The Hexadecimal digit display

This box displays a hexadecimal digit at a specified coordinate in the visible display area.



**Input signals:**

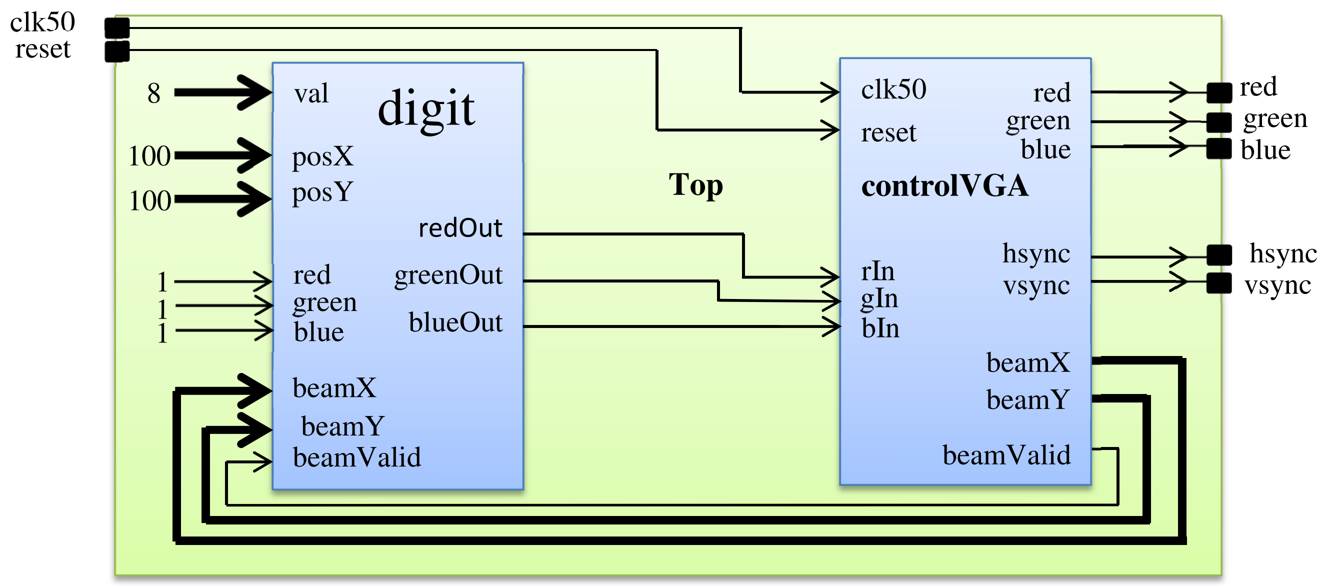
| | |
|---|---|
| val | Hexadecimal value on 4 bits |
| posX | X coordinate to display the digit. From 0 to 640. |
| posY | X coordinate to display the digit. From 0 to 480. |
| beamX | Current X position of the beam (refreshed pixel). From 0 to 640. |
| beamY | Current Y position of the beam (refreshed pixel). From 0 to 480. |
| beamValid | When high (‘1’) : the beam is in the visible area (640x480 area) |
| Red | Red color of the digit |
| Green | Green color of the digit |
| Blue | Blue color of the digit |

**Ouput signals:**

| | |
|---|---|
| redOut | Red signal for the beam (refreshed pixel) |
| greenOut | Green signal for the beam (refreshed pixel) |
| blueOut | Blue signal for the beam (refreshed pixel) |

## Architecture to design by using the structural view

You have to connect the 2 boxes in order to display the "8" value on the screen at coordinates (100, 100). The following schematic shows you how to connect the boxes:

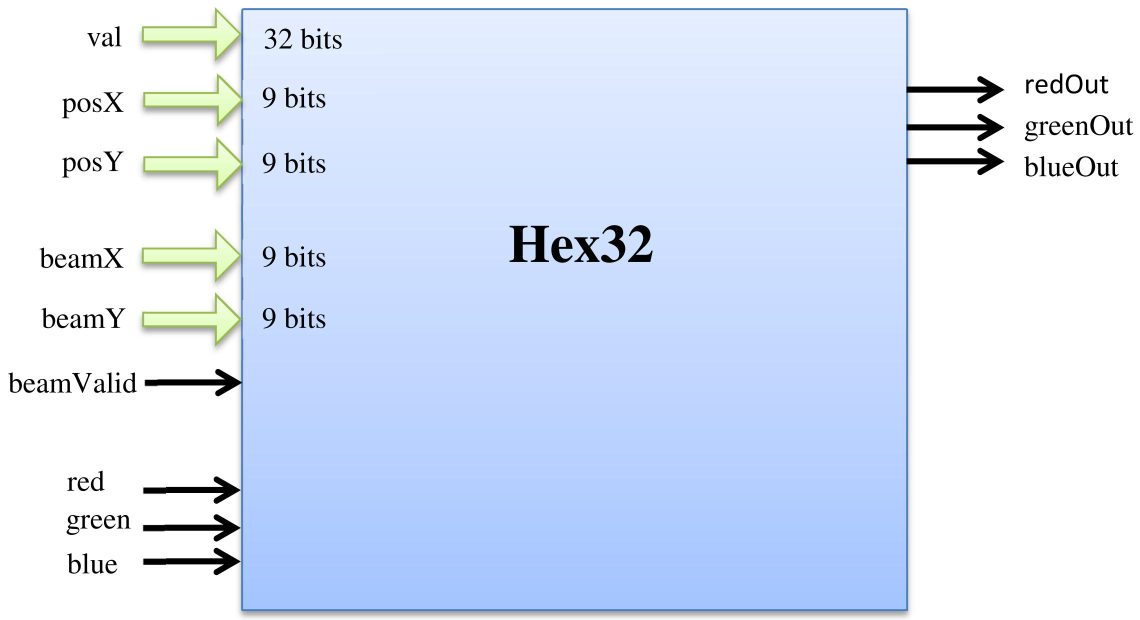


Create the top box and use ***port map/component*** keywords to instanciate digit and controlVGA boxes. To help you, you can use the example in course (slide : ***structural view***).

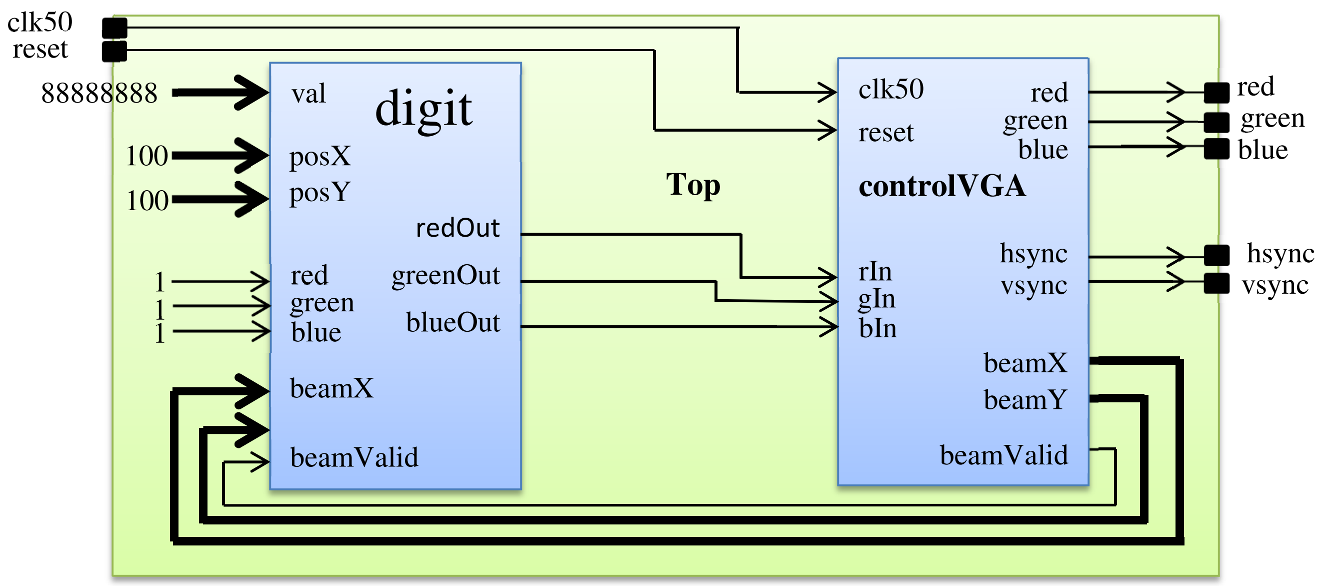
1) Create a project
2) Copy vhdl files ***digit*** and ***controlVGA*** in your project
3) Create ***Top*** file
4) Describe the schematic above with a structural view
5) Create a testBench to test the ***Top*** box (2 inputs: reset and clk50, 5 outputs: red, green, blue, hsync, vsync). An example of test bench is given in the course.
6) Check  the red/green/blue/hsync/vsync signals
7) Once the simulation validated: connect the interface signals of top to the matching FPGA pins (the site location of each signals are written on the board. The 50MHz clock is connected to C9. The reset will be connected on a switch (L13 for example)).
8) Create a bit file
9) Download the design on the board
10) Connect a VGA screen on the board and a "8" character should appear on the screen.
11) How many LUTs (CLB) does your design use?

## Create a 32 bits number display

With the digit box and the structural view create a box (called ***Hex32***) which is able to display a 32 bits number in hexadecimal (8 digit). For this part, you will have to describe on a paper the Hex32 box. This box will have to use 8 ***digit*** boxes.



You will also have to describe the ***top*** box which includes the ***hex32*** and the ***controlVGA*** boxes.



1) Describe on a paper the schematic of your ***Hex32*** box. This box has to use the ***digit*** box.
2) Create a new project
3) Copy vhdl files ***digit*** and ***controlVGA*** in your project
4) Create ***Hex32*** file. This file will be the VHDL description of your schematic. It will instanciate digit boxes by using ***component*** and ***port map*** keywords.

5) Create a testBench to test the ***Top*** box (2 inputs: reset and clk50, 5 outputs: red, green, blue, hsync, vsync). An example of test bench is given in the course.
6) Check  the red/green/blue/hsync/vsync signals
7) Once the simulation validated: connect the interface signals of top to the matching FPGA pins (the site location of each signals are written on the board. The 50MHz clock is connected to C9. The reset will be connected on a switch (L13 for example)).
8) Create a bit file
9) Download the design on the board
10) Connect a VGA screen on the board and a "88888888" character should appear on the screen.
11) How many LUTs (CLB) does your design use?

## The combinatory logic:

### Hack it!

Instead of using structural view to describe the ***Hex32*** box you will have to modify the ***digit*** file to create the ***Hex32*** box. So you will have to understand how works the ***digit*** box and modify it (hack it) in order to display a 32 bits number instead a 4 bits number.

1) Describe on a paper the schematic of the initial ***Digit*** box. This step will prove if you understand the VHDL code of ***digit***.
2) Describe on a paper the schematic of ***Hex32*** box.
3) Create a new project
4) Copy vhdl files ***digit*** and ***controlVGA*** in your project
5) Modify the ***digit*** file and write in VHDL the description of your Hex32 schematic. It will NOT instanciate ***digit*** boxes NOR use ***component*** and ***port map*** keywords.
6) Create a testBench to test the ***Top*** box (2 inputs: reset and clk50, 5 outputs: red, green, blue, hsync, vsync). An example of test bench is given in the course.
7) Check  the red/green/blue/hsync/vsync signals
8) Once the simulation validated: connect the interface signals of top to the matching FPGA pins (the site location of each signals are written on the board. The 50MHz clock is connected to C9).
9) Create a bit file
10) Download the design on the board
11) Connect a VGA screen on the board and a "88888888" character should appear on the screen.
12) How many LUTs (CLB) does your design use?

## Sequential logic

### Let's count

Instead of displaying a constant value ("88888888") we are going to display a 32 bits counter. This counter will be describe in the top file and with the data flow description.

The counter will use the clk50 signal as clock and the reset signal as reset (restart to "00000000").

1) Describe on a paper the schematic of ***Top*** box which includes the ***Hex32/controlVGA*** boxes and the 32 bits counter.
2) Create a new project
3) Copy vhdl files ***Hex32*** and ***controlVGA*** in your project
4) Modify the ***Top*** file and add the 32 bit counter description in data flow.
5) Create a testBench to test the ***Top*** box (2 inputs: reset and clk50, 5 outputs: red, green, blue, hsync, vsync). An example of test bench is given in the course.
6) Check  the behavior of your counter signals
7) Once the simulation validated: connect the interface signals of top to the matching FPGA pins (the site location of each signals are written on the board. The 50MHz clock is connected to C9).
8) Create a bit file
9) Download the design on the board
10) Connect a VGA screen on the board and … what happens?
11) How many LUTs (CLB) does your design use?

## Time control.

The counter seems to be too fast. In this part you will have to create a counter which adds 1 every second (in other words: a second counter). You will have to add another counter (the 50MHz cycle counter) which will enable the first counter (the display (or he second) counter) to add "1". Every counter must be connected on the 50MHz clock.

Questions:

How much 50MHz clock rising edges there are in 1 second?

What will be the maximum value of the 50Mhz cycle counter in order to send one signal (pulse) every second?

What kind of signal you have to add to your first counter? (have a look to the counters described in the course)

1) Describe on a paper the schematic of ***Top*** box which includes the ***Hex32/controlVGA*** boxes and the 32 bits second counter.
2) Create a new project
3) Copy vhdl files ***Hex32*** and ***controlVGA*** in your project
4) Modify the ***Top*** file and add the 32 bit second counter description in data flow.
5) Create a testBench to test the ***Top*** box (2 inputs: reset and clk50, 5 outputs: red, green, blue, hsync, vsync). An example of test bench is given in the course.
6) Check  the behavior of your second counter.
7) Once the simulation validated: connect the interface signals of top to the matching FPGA pins (the site location of each signals are written on the board. The 50MHz clock is connected to C9).
8) Create a bit file
9) Download the design on the board

10) Connect a VGA screen on the board and … what happens?
11) How many LUTs (CLB) does your design use?

## Let's move.

The goal of this last question is to dynamically move the counter on the screen on X or Y axis or both ones. How are you going to do that?

1) Describe on a paper the schematic of ***Top*** box which includes the ***Hex32/controlVGA*** boxes, the 32 bits second counter and something that is able to move the counter on the screen.
2) Create a new project
3) Copy vhdl files ***Hex32*** and ***controlVGA*** in your project
4) Modify the ***Top*** file and add the 32 bit second counter description and everything else in data flow.
5) Create a testBench to test the ***Top*** box (2 inputs: reset and clk50, 5 outputs: red, green, blue, hsync, vsync). An example of test bench is given in the course.
6) Check  the behavior of your second counter.
7) Once the simulation validated: connect the interface signals of top to the matching FPGA pins (the site location of each signals are written on the board. The 50MHz clock is connected to C9).
8) Create a bit file
9) Download the design on the board
10) Connect a VGA screen on the board and … what happens?
11) How many LUTs (CLB) does your design use?

## Make your own game

Now you should be able to create your own game from scratch.

1) Find a game idea (this step must be short. Some ideas: pong, brick breaker, game of life, memory game, reflex game, snake, road frog, etc…)
2) Cut the design of your game in small parts. Don't try to code the whole game in 1 time: **IT WILL NOT WORK!** Design it and check it step by step.  For example: if you want to design a pong game, first code the racket and check it, next the ball on 1 and 2 directions, next the collision between the racket and the ball, next add some direction to the ball, next add some speed difficulty level, etc…..
3) For each step, draw the schematic of your design. **WE WILL CHECK THEM!!!!**