ESN Project - I2C enabled Sopc + accelerometer + 💻 VGA ✳️

note: for the best viewing experience, view this document in vscode with the markdown preview enhanced by Yiyi Wang extension installed, otherwise open the PDF version found here

you may need to set this setting in the extension's settings (done through the GUI):

```
plantuml server:
https://kroki.io/plantuml/svg/
```

to start cygwin from powershell:

```
& 'C:\intelFPGA\18.1\nios2eds\Nios II Command Shell.bat'
```

to generate bsp:

```
nios2-bsp hal ./software/bsp/ ./*.sopcinfo
```

to generate makefile:

```
nios2-app-generate-makefile --app-dir ./software/app --bsp-dir ./software/bsp --elf-name maion.elf --src-files ./sofware/app/main.c
```

both (one liner):

```
nios2-bsp hal ./software/bsp/ ./*.sopcinfo && nios2-app-generate-makefile --app-dir ./software/app --bsp-dir ./software/bsp --elf-name maion.elf --src-
```

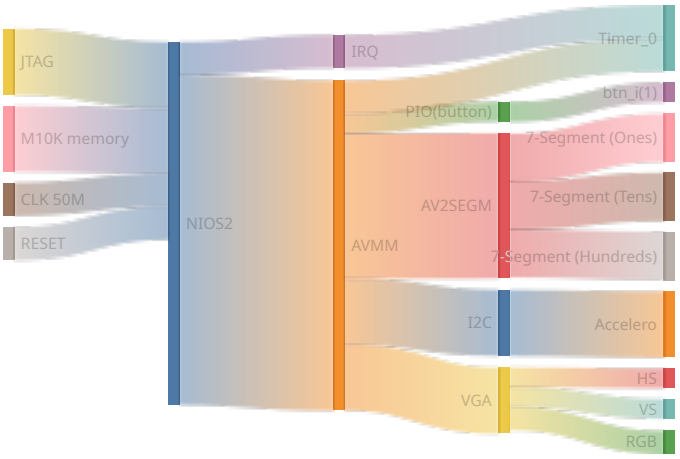
Introduction

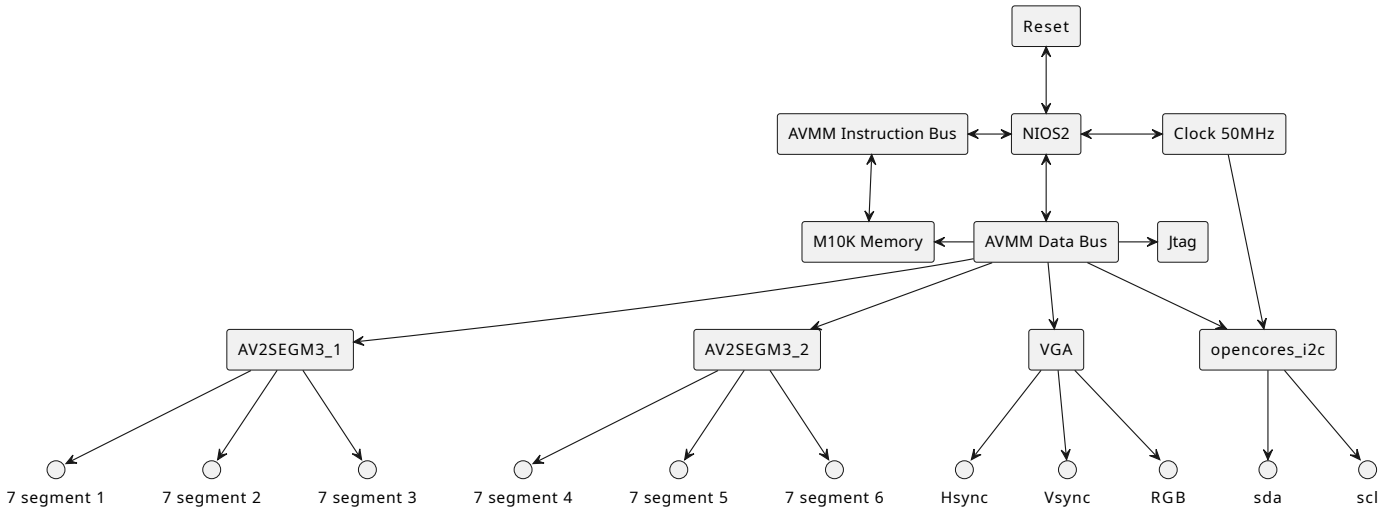
This project aims to add an I2C interface using an open cores IP block over AVMM, in order to communicate with an accelerometer.

This version has an entire VGA driver built from scratch added to it do display the data in the form of a 2D crosshair on a screen (640x480 at 30 fps).

System architecture

Similar to the other version of this project, with the added VGA hardware.





Progress

I won't document the things i already documented in the main(old) branch already, so i'll only show what i've change since

First of all and the most obvious, a VGA interface has been made from scratch. It outputs 640x480 12 bit RGB video at 30 Fps from a 50 Mhz clock (might be 60 Fps but i have conflicting info on this because in theory it should work at 25.125Mhz but this frequency doesn't work right, whereas just 50 does? moving on...).

This interface does double duty: it both manages the 12 bit vga port, but it also takes in a 2D coordinate through AVMM, this coordinate is used in it's drawing combinatorial logic to place the red "crosshair" in the white "box", see here

On the NIOS2 side, the sampling speed has been made faster and a simple low pass filter (average) has been implemented to make the crosshair less "jittery". The crosshair is managed in such a way that the values are locked for one frame at a time, the VHDL only updates it once the frame is finished to avoid tearing.

Conclusion

VGA is interesting, it's a good experience to learn about how screens work in general, especially in regards to their timing (blanking, porching...). It provides a pretty good visual interface of an FPGA system if you can get it to work reliably.