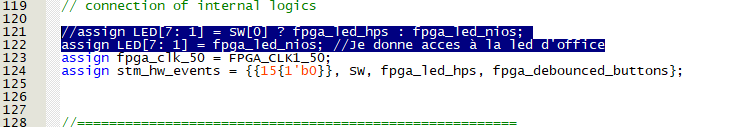
**LINGI 2315 - Homework 6 - My Nios App 2 P1**

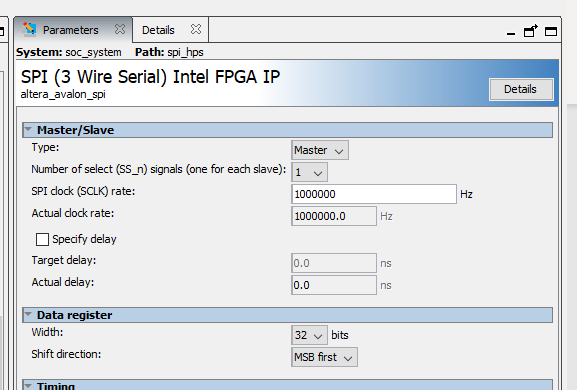
|  |  |  |  |
| --- | --- | --- | --- |
| Name | Delcoigne Ben | Noma | 3877 1700 |

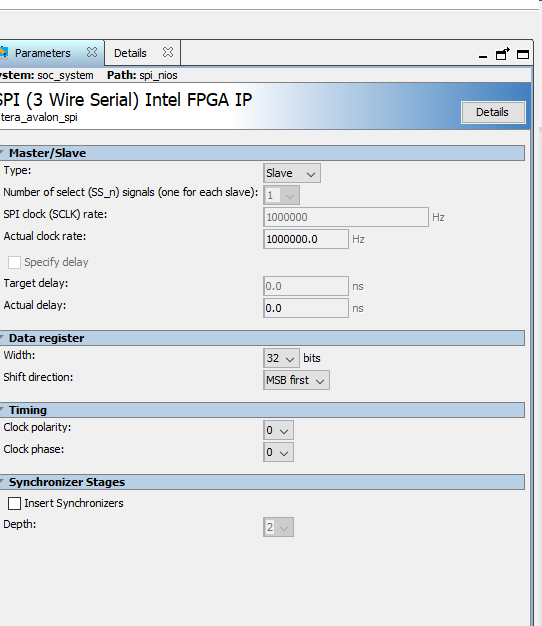
Description of the hardware with relevant screenshots of your SystemVerilog code and Qsys schematics

I give access to the LED all the tipe to the nios part:



I also switched the master-slave setup:





After doing so, I compiled the system again and generated a hdl\_0.h file.

**LINGI 2315 - Homework 6 - My Nios App 2 P2**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Delcoigne Ben | Noma | 38771700 |

Description of the software on Nios with relevant screenshots of your code

In the NIOS part, I am reading what is in the SPI bus:



I am basically just reading the value from the nios (printing it out in the console), and then applying a simple function that displays the value on the LEDS. Not much more was done in the nIOS part.

**LINGI 2315 - Homework 6 - My Nios App 2 P3**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Delcoigne Ben | Noma | 38771700 |

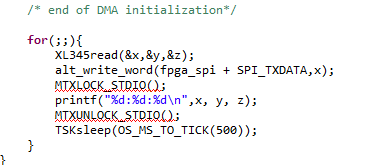
Description of the software on HPS with relevant screenshots of your code

For the HPS part, I had to first initialize the accelerometer. This was done the same way as for homework 3: init the dma, init the i2c, and then use functions XL435\_init and read in order to use the accelerometer.

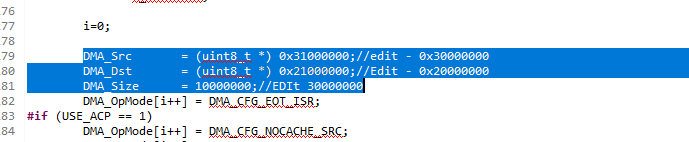
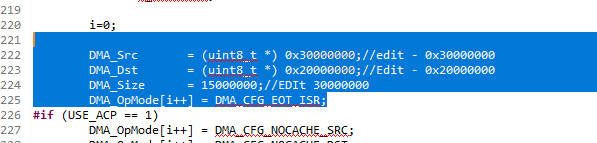
I created a task task\_gsensor that reads the gsensor and sends the value into the I2C bus.

I also adapted the task DMA (which is freed with the button semaphore) in order to do two transfers:

Gsensor:



For the file transfer (task dma), I used the code from myapp\_dma and copied it two times with different values for the transfer:

Note about the ACP:

When enabling it, the speed is much slower. This is due to the shared L2 cache between the two cores. (L1 is not shared). Indeed, when ACP is enabled, it checks consistency between the two core’s memory which takes time.

