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

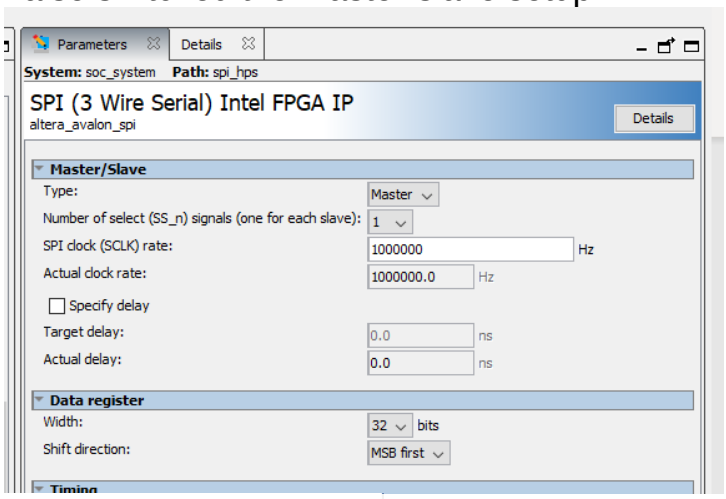
Name	Delcoigne Ben	Noma	3877 1700

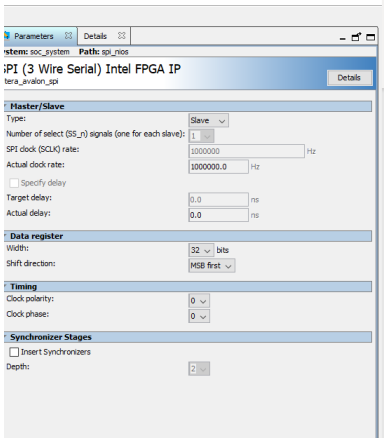
Description of the hardware with relevant screenshots of your SystemVerilog code and Qsys schematics

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

```
119     // connection of internal logics
120
121     //assign LED[7: 1] = SW[0] ? fpga_led_hps : fpga_led_nios;
122     assign LED[7: 1] = fpga_led_nios; //Je donne acces à la led d'office
123     assign fpga_clk_50 = FPGA_CLK1_50;
124     assign stm_hw_events = {{15{1'b0}}, SW, fpga_led_hps, fpga_debounced_buttons};
125
126
127
128     //=======================================================
```

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:

```
#define TRUE    1
#define FALSE   0

int main()
{
    //unsigned int value = 0;
    int serial = 1;

    printf("NIOS Version 13 turns on  \r\n");

    while(1) {
//      IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_NIOS_BASE, value++);//This is just the le
//
//
//      if(serial >=128){
//          serial=1;
//      }
//      else{
//          serial = serial<<1;
//          IOWR_ALTERA_AVALON_SPI_TXDATA(SPI_NIOS_BASE, serial);
//      }
        serial = IORD_ALTERA_AVALON_SPI_RXDATA(SPI_NIOS_BASE);
        printf("Recieving value in NIOS: %d\n", serial);
        if(serial<0){
            serial = -serial;//Absolute value
        }
        if(serial<100){
            IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_NIOS_BASE,0x1);
        }
        else if(serial<200){
            IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_NIOS_BASE,3);
        }
        else if(serial<500){
            IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_NIOS_BASE,15);
        }
        else if(serial<700){
            IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_NIOS_BASE,31);
        }
        else if(serial<1000){
            IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_NIOS_BASE,63);
        }


        else{
            IOWR_ALTERA_AVALON_PIO_DATA(LED_PIO_NIOS_BASE,0);
        }
        IOWR_8DIRECT(ONCHIP_MEMORY2_0_BASE, 0x20000000,0xFF);
        IOWR_8DIRECT(ONCHIP_MEMORY2_0_BASE, 0x21000000,0xFF);
        usleep(500000);//Changed slp time
    }
}
```

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:

```
    /* end of DMA initialization*/

    for(;;){
        XL345read(&x,&y,&z);
        alt_write_word(fpga_spi + SPI_TXDATA,x);
        MTXLOCK_STDIO();
        printf("%d:%d:%d\n",x, y, z);
        MTXUNLOCK_STDIO();
        TSKsleep(OS_MS_TO_TICK(500));
    }
}
```

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

```
76
77          i=0;
78
79          DMA_Src       = (uint8_t *) 0x31000000;//edit - 0x30000000
80          DMA_Dst       = (uint8_t *) 0x21000000;//Edit - 0x20000000
81          DMA_Size      = 10000000;//EDIt 30000000
82          DMA_OpMode[i++] = DMA_CFG_EOT_ISR;
83  #if (USE_ACP == 1)
84          DMA_OpMode[i++] = DMA_CFG_NOCACHE_SRC;
219
220           i=0;
221
222         DMA_Src       = (uint8_t *) 0x30000000;//edit - 0x30000000
223         DMA_Dst       = (uint8_t *) 0x20000000;//Edit - 0x20000000
224         DMA_Size      = 15000000;//EDIt 30000000
225         DMA_OpMode[i++] = DMA_CFG_EOT_ISR;
226 #if (USE_ACP == 1)
227         DMA_OpMode[i++] = DMA_CFG_NOCACHE_SRC;
```

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.