System on Chip: Class Report 4

Noel Sengel and John Westbrook

November 25, 2023

Introduction

In this class report, the purpose was to create an Inclination Sensor with Pon Chu's Microblaze MCS package and the various files with the Accelerometer. The hardware side in Verilog was quite simple, it involved us following Chu's instructions and installing the correct files and constraints. We then installed the correct CPU package. We then installed the current GPIO modules that correlate with the Accelerometer. on board. We dove into the IO reads and writes and then moved into the C+= side of the code. There we wrote high level functions that would take data from the Accelerometer, and determine the angle of the board.

Experimental Plan

You can see in Listing 1 how we wrote the C++ code and used the hardware code written in SystemVerilog. After testing, I was able to see that the code worked correctly and the board was able to tell where it was.

Analysis

The Following code shows the Vitis C++ code that was included in this project. After compiling on the board, you can see what Chu intended with this board. In this GitHub link provided below, there is a video of our board working.

Code

Here is the GitHub repo with our modules: https://github.com/NoelSengel/Inclination-Sensing

Listing 1: Main Logic of Inclination Sensing

```
int main() {

   //uint8_t id, ;
   //SpiCore spi_p;
   SpiCore *spi_p = &spi;
   GpoCore *led_p = &led;
   int i, n;
   uint8_t dp;
   const uint8_t RD_CMD = 0x0b;
   const uint8_t PART_ID_REG = 0x02;
   const uint8_t DATA_REG = 0x08;
   const float raw_max = 127.0 / 2.0;  //128 max 8-bit reading for +/-2g
   const uint8_t SSEG_0[] = {0xff,0xff,0xff,0b1000000,0xff,0xff,0xff,0xff
      };
```

```
const uint8_t SSEG_90[] = {0xff,0xff,0b1000000,0b0011000,0xff,0xff,0xff
   ,0xff};
const uint8_t SSEG_180[] = {0xff,0b1000000,0b0000000,0b1111001,0xff,0
   xff,0xff,0xff};
const uint8_t SSEG_270[] = {0xff,0b1000000,0b1011000,0b0100100,0xff,0
   xff,0xff,0xff};
sseg.set_dp(0x00);
int8_t xraw, yraw, zraw;
float x, y, z;
spi_p->set_freq(400000);
spi_p->set_mode(0, 0);
// check part id
while (1) {
   spi_p->assert_ss(0);    // activate
   spi_p->transfer(RD_CMD);  // for read operation
   spi_p->transfer(PART_ID_REG);  // part id address
   spi_p->deassert_ss(0);
    // read 8-bit x/y/z g values once
   spi_p->assert_ss(0);    // activate
   spi_p->transfer(RD_CMD);  // for read operation
   spi_p->transfer(DATA_REG);  //
   xraw = spi_p->transfer(0x00);
   yraw = spi_p->transfer(0x00);
   zraw = spi_p->transfer(0x00);
   spi_p->deassert_ss(0);
   x = (float) xraw / raw_max;
   y = (float) yraw / raw_max;
   z = (float) zraw / raw_max;
   uart.disp("x/y/z␣axis␣g␣values:␣");
   uart.disp(x, 3);
   uart.disp("␣/␣");
   uart.disp(y, 3);
   uart.disp("␣/␣");
   uart.disp(z, 3);
   uart.disp("\n\r");

   //const uint8_t SSEG_PTN[]={0xff,0xf9,0x89,0xff,0xff,0xff,0xff,0xff
      };
   //base_addr = core_base_addr;
   //write_8ptn((uint8_t*) HI_PTN);
   //set_dp(0x02);

    //0 degrees
    if((x >= -0.005) && (x <= 0.005) && (y >= -0.005) && (y <= 0.005)){
      sseg.write_8ptn((uint8_t*) SSEG_0);
      led_p->write(1, 1);
      led_p->write(0, 2);
      led_p->write(0, 3);
      led_p->write(0, 4);
      sleep_ms(100);
    }
   // 90 degrees
   if((x >= 0.005) && (x <= 0.020) && (y >= -0.005) && (y <= 0.005)){
     sseg.write_8ptn((uint8_t*) SSEG_90);
```

```
        led_p->write(1, 2);
        led_p->write(0, 1);
        led_p->write(0, 3);
        led_p->write(0, 4);
        sleep_ms(100);
      }
      // 180 degrees
      if((x >= 0.005) && (x <= 0.020) && (y >= 0.005) && (y <= 0.020)){
        sseg.write_8ptn((uint8_t*) SSEG_180);
        led_p->write(1, 3);
        led_p->write(0, 1);
        led_p->write(0, 2);
        led_p->write(0, 4);
        sleep_ms(100);
      }
      // 270 degrees
      if((x >= -0.005) && (x <= 0.005) && (y >= 0.005) && (y <= 0.020)){
        sseg.write_8ptn((uint8_t*) SSEG_270);
        led_p->write(1, 4);
        led_p->write(0, 1);
        led_p->write(0, 2);
        led_p->write(0, 3);
        sleep_ms(100);
      }
    } //while

} //main
```