Exercise 11.

Measurement of programmable peripheral units

Goal of the measurement

- Digital design knowledge
- Verilog knowledge
- Xilinx ISE (Exercise 3 & 10)
- SPI & UART communication standards (Serial Communication Standards documentation)

Goal of the measurement

The goal of this lab is (1) to study and/or implement the UART and SPI communication protocols, (2) to implement a typical peripheral interface (3) get more experience about CAD aided hardware design.

Preparing for the measurement

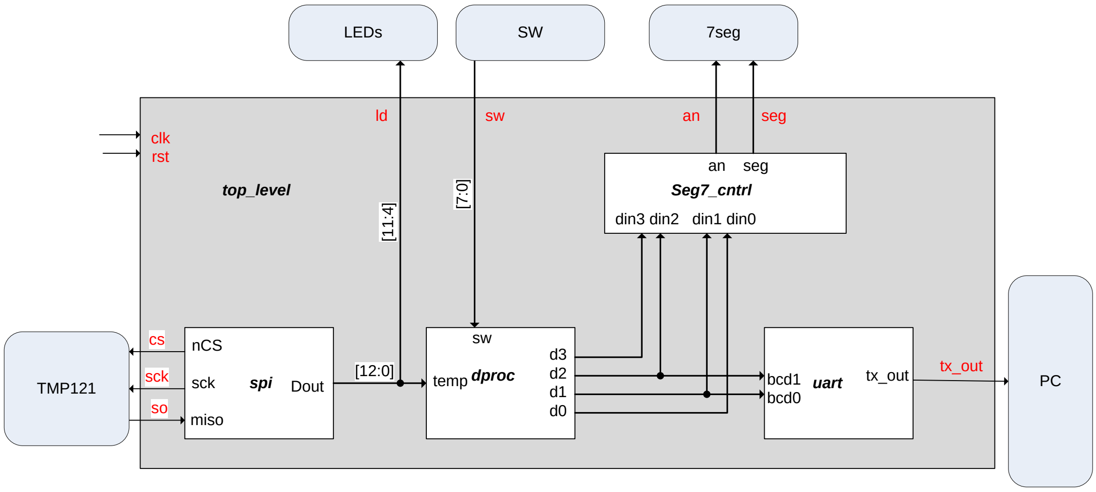
This measurement is based on earlier exercises. Scrutinize what you have learned in “Digital devices basics” and “Implementation and analysis of sequential networks”!

Read the papers available on the homepage: “Serial Communication Standards”, “Design of an SPI receiver”

Check the measurement tasks and the test questions!

Measurement tasks

The block diagram of the complete hardware to implement is depicted below. All of the modules use the system clock and reset input.

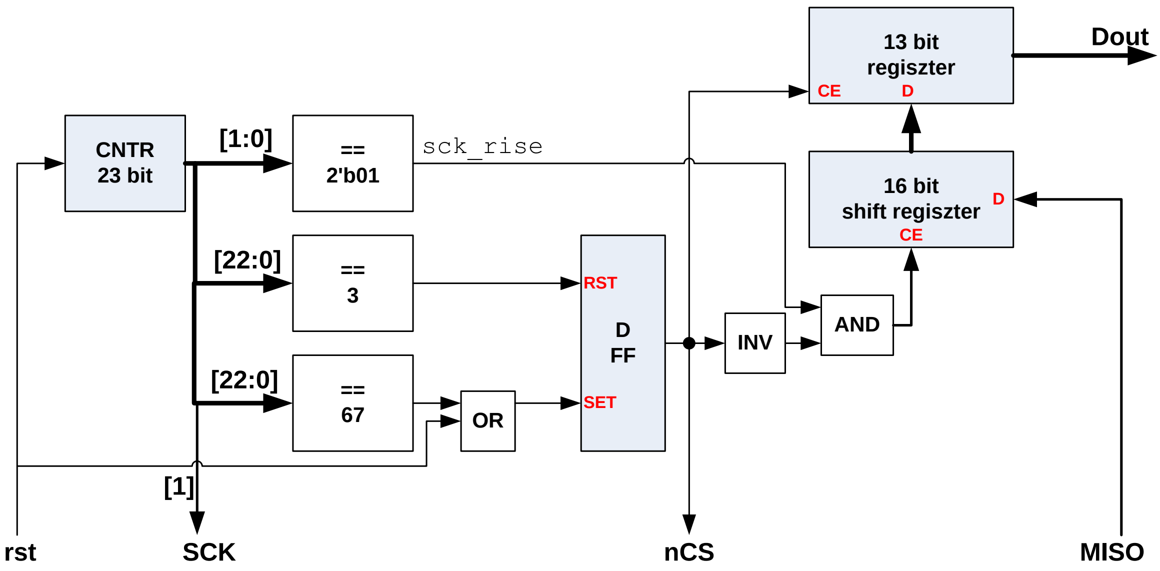


The following skeleton files can be downloaded from the homepage:

- In *top_level* module every sub-module but the UART is instantiated.
- *Seg7_cntrl*: The 7-segment display controller module you have designed earlier. The only improvement is that beside the BCD codes, this controller is able to display the minus sign on a digit (if the input is 13d) or to switch the digit off (for any other non-BCD input). The decimal point for the segment *din1* is always on, for all the other segments, it is always off. The internal enable signal is generated in a more efficient way as before. You have nothing to modify in this module, but examine the its new features and modifications.
- *spi*: the skeleton for the SPI receiver to implement. Initially *nCS* = 1, *sck* = 0, *Dout* = 13'b0. This should be modified, and Dout should contain the 13 bit temperature value. The integer part will be displayed on the LEDs.
- The *tb_spi_temp* is a test bench for the SPI module. During the measurement, you hace to examine and use this. (There is nothing to modify.)
- The input for the *dproc* module is the temperature and the value of the switches. The output is a 4 digit BCD value. Depending on sw[7], the output represents directly sw[6:0] or temperature-sw[6:0]. During the measurement, you have to examine this module and to do a little modification.
- The *pins.ucf* contains the pinout for our hardware. There is nothing to modify.
- The uart is added to the project as a black-box module.
- *cs.zip*: a chip scope configuration file. Initially, do not add this to your project!

1. SPI receiver: implementation, simulation and test

Based on the description made in the paper "Design of an SPI receiver", selecting SCK for 4 MHz-re, and considering that the system clock is 16 MHz, the temperature conversion time is 130 ms, and 13 of 16 transferred data bits are relevant, we get the following block diagram:



1.1. Examine the block diagram! Is the specification of TMP121 fulfilled? How long is one communication cycle? What are the 3 comparators for? Is the Dout output always consistent?

1.2. Implement the SPI module and check its functionality with simulation! You should understand the structure of the given test bench file! How is the temperature sensor simulated? How can you see on the simulation waveforms, wether your module is functioning well?

1.3. Generate and download the bit file, check the functionality without offset temperature (sw[6:0]=0)! Set up 42 deg. offset! What have you experienced? Supplement the dproc module in order to correct this error?

2. Optional task

2.1. Evaluate the binary-BCD converter implemented in the *dproc* module! What is the used conversion algorithm? How many clock cycles does a conversion take? When is a new conversion started? What is the role of the following registers: *data_old, data_out, data_conv* és *data_high?* What are their values before and after a conversion?

Test questions

1. What are the properties of the SPI interface? What signals are required for communication and what is the role of these signals?
2. Draw a time diagram of the TMP121 communication.
3. How many wires are needed to use 4 peripherals on the SPI bus, what are they?
4. What is the fixed point number representation? What is the value of each bit?
5. In what format does the TMP121 send temperature data? What temperature does the hexadecimal value 0x0C88 correspond to?
6. Write the value -42h in 2's complement binary 8-bit integer format. What decimal value does the bit pattern correspond to if it is considered a signed, fixed-point value with two fractional bits?

7. Give the Verilog code for a 16-bit shift register with an enable input which shifts to the left. The inputs to the module are clock (clk), enable (en); the output is the current state of the shift register (shr).
8. Give the Verilog code of an 8-bit shift register with an enable input, which shifts to the right. The inputs to the module are: clock (clk), enable (en); the output is the current state of the shift register (shr).
9. Give the Verilog code of a 20-bit up-counter. The inputs to the module are clock (clk), reset (rst); the output is the current state of the counter (cntr).
10. Specify (with an assign statement) the Verilog code to calculate the absolute value of a 14-bit two's complement value. The input signal shall be named data_2comp and the absolute value signal shall be named data_abs. The latter signal is also declared.