Übung Echtzeitsysteme WS 2013 / 2014
Atmel AVR

Philipp Heise, Christian Buckl

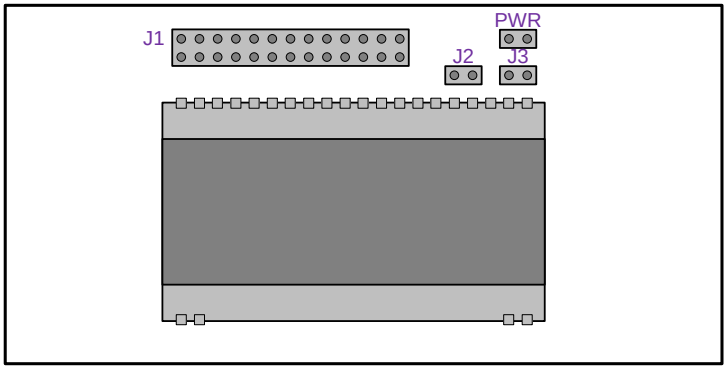
Exercise 0 Serial Peripheral Interface - SPI protocol and communication

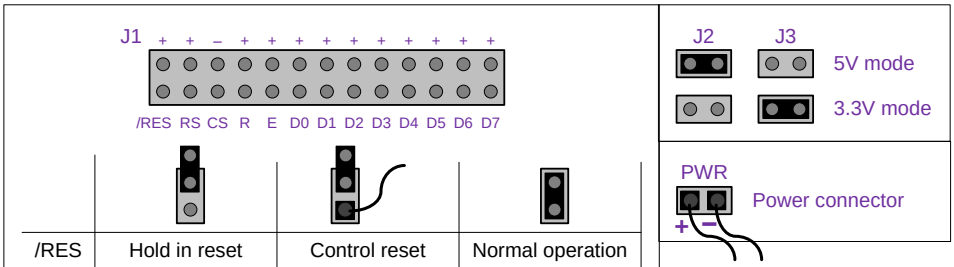
In the following exercise we want to use the SPI protocol to communicate with other hardware or sensors.

- Read about the SPI procotol on the internet (http://de.wikipedia.org/wiki/Serial_Peripheral_Interface, http://en.wikipedia.org/wiki/Serial_Peripheral_Interface).
- What does master and slave mean in the SPI context?
- Why is there a special wire for the clock? How can the clock signal be interpreted by the slaves - is there more than one plausible setting?
- Besides the clock what need the master and slave to agree upon if bytes are transmitted?
- How can one master address more than one slave?
- Read chapter 19 in the ATMega168 manual. How can you set all the previously mentioned settings? How is data transmitted and how can one find out that the transmisson has finished?

Exercise 1 SPI LC-Display

In this exercise we will communicate with an SPI compatible display. The schematic of the display and its carrier board are shown below.





/RES	Hold in reset	Control reset	Normal operation
RS	–[1]	Control Cmd/Data	–[1]
CS	Chip not selected	Control CS	Chip selected
R	–[1]	Control Read/Write	SPI mode
E	–[1]	Control Enable	SPI mode
D0	–[1]	Control D0	SPI / 4 Bit mode
D1	–[1]	Control D1	SPI / 4 Bit mode
D2	–[1]	Control D2	SPI / 4 Bit mode
D3	–[1]	Control D3	SPI / 4 Bit mode
D4	–[1]	Control D4	SPI mode
D5	–[1]	Control D5	SPI mode
D6	–[1]	Control D6	–[1]
D7	–[1]	Control D7	–[1]

[1] Do not use

- Connect the display to the STK500. The `PWR` connector has to be connected to `VTG` and `GND` on STK500 (respecting polarity). `D6` (`CLK`) has to be connected to `PB5` (`SCK`) and `D7` (`SI`) to `PB3` (`MOSI`). Further we connect `RS` to `PB1` and `/RES` to `PD2` to be able to reset the display controller upon startup (needed for reliable initialization). With `/RES` we can reset the display controller. The `RS` pin allows us to tell the display controller if the transfered data has to be interpreted as a command or if it should be displayed.
- Why is it not necessary to connect the slave output to the AVR? What are the implications of this?
- Edit the `spi.c` file and setup the control registers for the AVR in the `spi_init` routine. The AVR needs to be the master, MSB first and `CPHA,CPOL` should both be zero. The `SCK` rate should be set to $\frac{1}{4}$ of the oscillator frequency (no double rate).

- What does the function `_lcd_command_mode( uint8_t cmd )` in `lcd.c` do? Implement the functions `void lcd_set_cursor(uint8_t pos)` and `void lcd_showchar( char c )` in `lcd.c`. Keep in mind to respect the maximum timings and command codes provided in the datasheet of the display.
- Modify `main.c` to show some other text on the display. Find out how and why `fprintf` together with `stderr` can be used to show text on the display. Can you change it to `stdout` and use `printf`?