Tipps fürs Programmieren

Student Group

First Name	Surname	Matrikel Nr.

Table of Contents

Tips for Programming

- If the task involves hardware-software co-design, the creation of the software or the software system development can already begin in parallel with the schematic design or directly afterwards.
- First, think about
  - **what** the software has to do (higher-level tasks),
  - and in **which sequence** it should do so
- Then you can consider
  - how these individual tasks can be assigned to **C functions**,
  - how the C functions depend on one another,
  - which interfaces the C functions require between each other (data type, value range, name)
- **Only then should you think about what the code looks like**. A look at the datasheets and application notes of the µControllers and chips can help here. These often already provide algorithms or code snippets.

Software System Design

If you do not yet know exactly how the software or the hardware to be used will be applied, the following tips should help:

- Do not search for the component only in German. The number of results increases many times over if you search in English. Linguee is recommended for translation.
- Also use image search if you do not know the exact term.
- If terms are unclear, the following search terms can also help: *`Arduino` + <English translation of the "thing" being searched for> + optionally `Project`*. Alternatively, AVR or `Atmel` can be used instead of Arduino. For C code, you can additionally search for "`c code`". Code is often found on GitHub, so adding `site:github.com` in Google may also be helpful.

Requirements for Evaluation

The Requirements for Programming contain notes on what the submitted code should look like.

Common Errors and Debugging

- Tips on debugging and common mistakes can be found in the Tips for Troubleshooting.
- Try to test your program after every small change whenever possible. If you change three things and only test afterwards, you will not know which change caused the issue!
- The 25 Most Common Programming Mistakes or as a paper and further 41 common mistakes

General

- A good introduction to embedded software development can be found in the book Sensor Networks in Theory and Practice - Successfully Realizing Embedded Systems Projects by colleagues Meroth and Sora. It explains how to get started in the field of software development embedded in hardware. From within the university network or via VPN, you can view it directly on Springer Link. Another good introduction can be found at Mikrocontroller.net.

- The hardware does not need to be fully completed in order to start programming. If you want to use a microcontroller from the ATmega family, you can already develop and test software with the MiniMEXLE, MEXLE2020, or Simulide.
- Tips for the
  programming of ATMEL chips
- If you need large tables, you should store the data in program memory (EEPROM) and not in data memory (SRAM). As a rule, the program memory is larger by a factor of 5 to 10.
- There is no if loop!
- If you search the web for solutions, note that Arduinos generally use C++ (e.g. file.cpp). In most cases this is not directly compatible. On the other hand, the concepts can still be adopted.
- for(x = 0 ; x < 400 ; x++) : If x is defined as an 8-bit integer, this loop will run forever…
- Variable types must be taken into account in calculations, otherwise c=a/b with int a=5 and int b=2 becomes 2. An explicit type cast helps here: c=(float)a/b

Serial Interfaces

- Programming an AVR chip via USB (if the chip supports this) is done using the tool FLIP
- If you want to control an external component via a microcontroller, the following must be considered: Check whether the external component triggers on the positive edge or on the negative edge. As a rule, this cannot be changed on the external component itself. This can be changed in software on the microcontroller side.
- If another I2C interface is required, you can find templates for this online.

Programming the ST7565 in the ERC 128 64 - 1 Display

- The
  ERC 128 64 - 1
  display with 128 pixels in the x-direction and 64 in the y-direction is divided into 8×8 sections. These 8×8 pixels are also called pages. In software, 132×65 pixels can be addressed, but output is only on 128×64 pixels.
- In the
  ST7565
  , each 8-bit vertical block is stored in one byte.
- The commands that can be used via SPI are described in the datasheet.
- Via SPI, the display can only be written to. Reading is not possible.
- Read the manufacturer's sample code and look for tutorials for the ST7565.
- Pay attention to the mounting orientation when controlling the display.

Using Ports

- The following must be observed if you want or need to use JTAG ports differently - e.g. PF4..7 on the ATMEGA16U4 - (JTAG ports = ports to which the programming hardware is connected): The JTAG ports cannot be used directly without further action. The JTAG interface must first be disabled using the following code:

```
MCUCR |=(1<<JTD);
MCUCR |=(1<<JTD);
```

  Important: The control register must be written twice.

- For time-critical output of consecutive bits (e.g. for controlling intelligent LEDs), interrupts must

definitely be used. It is also worthwhile to additionally rely on USART. With USART, the data to be sent is first written into the UDRn register and then transferred into the shift register.
- If you use an external oscillator or crystal, two ports are used for this (ports XTAL = “Crystal”). If you accidentally define these ports as outputs via DDR, the chip will no longer have a clock. That means this port assignment is the last thing the chip does... After that, it can only be revived via the debugger interface.

Simulide

Stepper Motor Driver

Various stepper motor drivers (such as the DRV8825) make it possible to drive a stepper motor by specifying the direction and pulsing the STEP pin. Such drivers are not available in Simulide. However, the following simulation at least allows this to be replicated: stepper.rar