SW-Developement for the GPS bycicle computer

Armin Schlegel,
Christian Eismann

May 10, 2011

1 The Toolchain

Hardware:

- Eagle - circuit layout and design

Software:

- virtualbox (Oracle) including a Debian Linux image
- GNU Compiler Collection (AVR-GCC)
- avrdude (programmer)
- Make
- (sp)lint - static code analysis
- ISP Programmer

2 Hardware

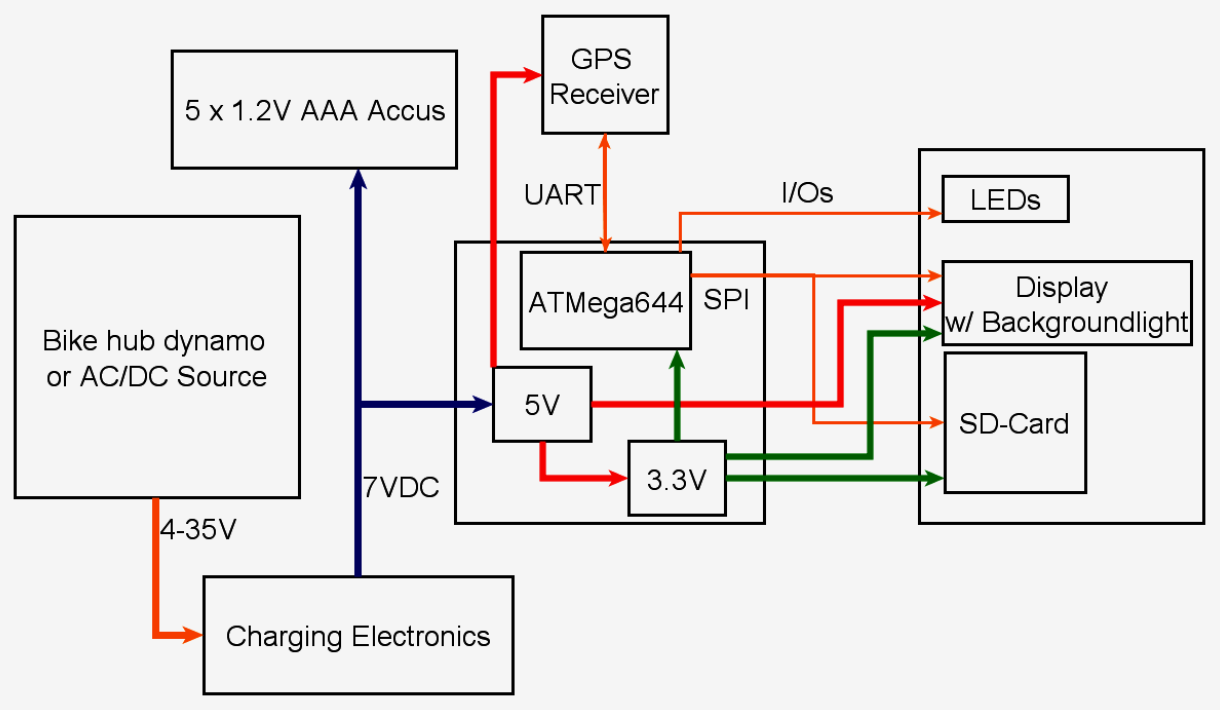
2.1 Requirements

- Processor supply voltage: 3.3V
- GPS receiver
- RS232 Debug/Communication Port (choosable via jumper)
- Usage of a graphic display
- SD card for data recording
- Mobile energy supply (chargeable)
- programmable via ISP (and JTAG)

2.2 Main components

- ATmega32L and later ATmega644p (3.3V)
- NL-552ETTL (GPS Receiver, 5V VCC, 3.3V RXD/TXD levels)
- EA-DOGL128-6 (Display, 3.3V VCC, SPI)
- YAMAICHI SD slot (3.3V VCC, SPI)
- MAX3221 (RS232 controller, 3.3V VCC)

2.3 Block Diagram



3 Software

3.1 Design topics

- Mapping features to various modules - The functionality is separated to different modules (e.g. GPS module, Interrupt module, ...)
- Running synchronous to GPS data receiving (triggered by USART interrupt)
  - no timing/scheduling problems
  - simple integration of the basic features
  - no OS required

  - but: no complex user interaction possible (e.g. via touch screen)

- no OS is used
  - integration too time consuming
  - functions have to be reentrant
  - balancing of tasks pretty time consuming and complex (detailed design necessary)

3.2 The NMEA/PUBX Protocol

- NMEA: simple (serial) ASCII protocol (standardized)
- PUBX: proprietary NMEA extension (used for configuration)

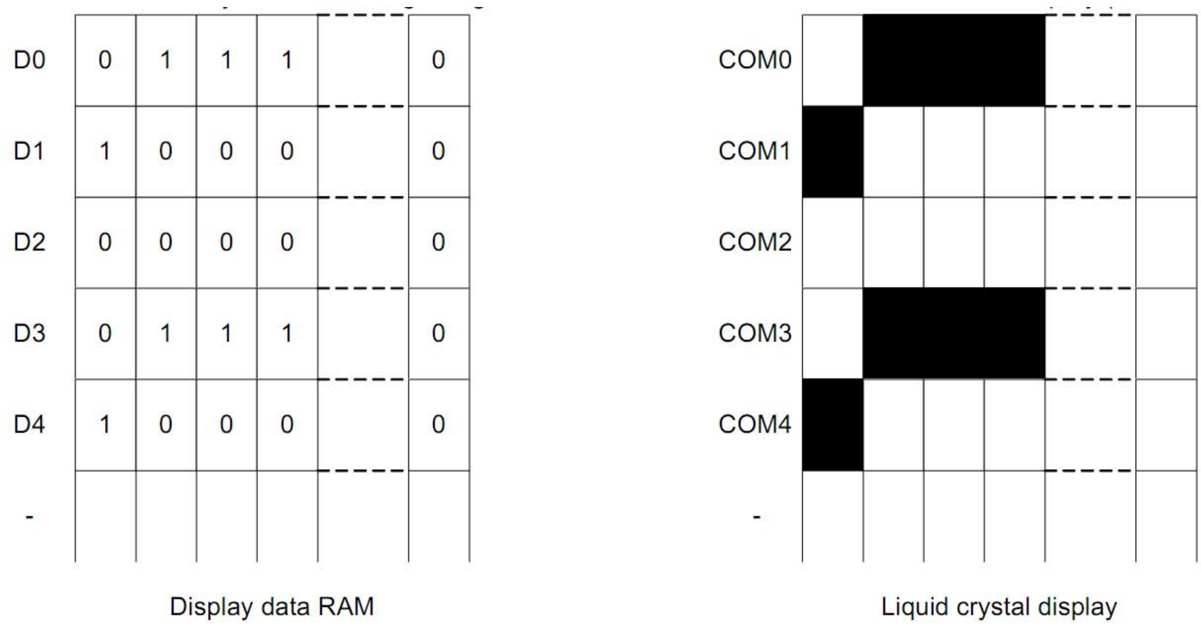
3.3 Components

3.3.1 SPI (Serial Peripheral Interface)

- simple SPI-driver
- used for communication between μC and SD card and the display

3.3.2 Display

Setting single pixels



D0	0	1	1	1		0
D1	1	0	0	0		0
D2	0	0	0	0		0
D3	0	1	1	1		0
D4	1	0	0	0		0
-						

Display data RAM

COM0		■	■	■		
COM1	■					
COM2						
COM3		■	■	■		
COM4	■					
-						

Liquid crystal display

- direct pixel access via display data RAM

- RAM is organized in pages
- total size of $(8pages * 8bit) * 132bit$ (actual only 128 bit, because of the display resolution of 128 x 64)
- internal SW buffer: sequential data structure (8 bit data type)
- positioning within the internal buffer for setting a pixel with coordinates X and Y: $INDEX = (Y * 8) + (X/8)$
- the actual bit is determined by : $BIT\_NR = Y\&0x07$ (determining the actual row within a memory page)
- Summary: the index of the data buffer represents the 'column' number within the data memory (of a single page), the result of a bit-wise AND operation with Y results in the actual row of the respective memory page

Listing 1: Example: display_putpixel() function

```c
/*
 * Set/Unset a single pixel on the display
 *
 * For choosing the correct entry within the data structure (disp_ram[])
 * first the concerning page of this pixel has to be determined. This
 * is done by dividing the Y coordinate by 8 (or better: do a right shift
 * of 3 bits). For choosing the entry in the array, the X coordinate multiplied
 * by 8 (or better: left shifted by 3) has to be added to the actual page number.
 * The exact bit that shall be set/unset is determined by using the bitmask
 * (y & 0x07). This selects the exact row of the respective memory page.
 *
 * Parameters:
 *    x   X coordinate of the pixel
 *    y   Y coordinate of the pixel
 */
void display_putpixel(unsigned char x, unsigned char y, uint8_t pixel_status)
{
   if (x < DISP_WIDTH && y < DISP_HEIGHT) {
      if (pixel_status == PIXEL_ON)
         disp_ram[(y >> 3) + (x << 3)] |= (1 << (y & 0x07));
      else
         disp_ram[(y >> 3) + (x << 3)] &= ~(1 << (y & 0x07));
   }
}
```

Drawing BMPs/Text

- BMPs: Converting BMPs into simple C-Arrays
- Text: using a 5x7 character set (organized in a simple one dimensional array)

3.3.3 LEDs

- simple IO access
- used as status indicator
- e.g. receiving of GPS data, recording of GPS data, ...

3.3.4 UART (Universal Asynchronous Receiver Transmitter)

- simple driver module
- used for communication between PC and μC
- or between GPS and μC

3.3.5 GPS

- initialization of the GPS receiver:
  - setting the baud rate (for synchronization with the μC
  - setting refresh rate to 1 per second (supports also 4)
  - selection of required data sets (RMC, GGA, VTG)
- splitting of NMEA data sets (',*' separator)
- storage into internal data structure

3.3.6 SDC/FAT16

- tiny open source library
- horrible code (e.g. huge amount of magic numbers, magic bit shifting with several side effects)

3.3.7 Touch screen

- used for start/stop data recording
- connected to MC via ADC

4 References

- http://www.lcd-module.de/pdf/grafik/dogl128-6.pdf, Electronic Assembly
- NL_u-blox5_Referenzmanual_06102008_571.pdf, u-blox 5 NMEA, UBX Protocol Specification
- http://www.atmel.com/dyn/resources/prod_documents/doc2593.pdf, ATmega644L datasheet
- http://focus.ti.com/lit/ds/symlink/max3221.pdf, MAX3221 datasheet
- http://www.national.com/ds/LM/LM3478.pdf, LM3478 Switching Regulator datasheet
- http://www.lcd-module.de/eng/pdf/zubehoer/st7565r.pdf, ST75565R display controller data sheet