Low Level AVR

Arduino? Nein danke!

Felix Queißner

shackspace

2020

Wer bin ich?

- Felix „xq“ Queißner
- Baujahr 1993
- Mit 12 angefangen, zu programmieren
- Mit 13 angefangen, Elektronik zu basteln
- Mache gerne Dinge mit Code und alten Computern

Worum geht's?

- Grundlagenwissen AVR-Programmierung vermitteln
- „Wie komme ich klar, wenn es mit der Arduino-IDE nicht mehr weiter geht?“
- „Awareness schaffen“ für Code Bloat
- **Nicht:** AVR/Arduino löten
- **Nicht:** C programmieren lernen

Outline

Wie kommen wir von hier ...

```
int button = getPressedButton();
if (button == 1) {
  digitalWrite(RELAIS_1_PIN, HIGH);
  delay(350);
  digitalWrite(RELAIS_2_PIN, LOW);
  digitalWrite(RELAIS_3_PIN, LOW);
  digitalWrite(RELAIS_4_PIN, LOW);
}
if (button == 2) {
  digitalWrite(RELAIS_1_PIN, LOW);
  digitalWrite(RELAIS_2_PIN, HIGH);
  delay(350);
  digitalWrite(RELAIS_3_PIN, LOW);
  digitalWrite(RELAIS_4_PIN, LOW);
}
if (button == 3) {
  digitalWrite(RELAIS_1_PIN, LOW);
  digitalWrite(RELAIS_2_PIN, LOW);
  digitalWrite(RELAIS_3_PIN, LOW);
```

… nach da?

```
1 int button = getPressedButton();
2 PORTD = 0x00;
3 if (button != 0) {
4   delay(350);
5   PORTD = (1 << (button + 2));
6 }
```

The Arduino way

- Schnell von „Idee“ zu „Es blinkt schon mal was“ kommen
- Single-Click-Solution
- Mit was man genau arbeitet ist eigentlich egal, es erfüllt alles den gleichen Zweck

Blink

- Lässt eine LED mit $\frac{1}{2}$ Hertz blinken
- Benutzt `setup()` und `loop()`

```
1  void setup() {
2    pinMode(LED_BUILTIN, OUTPUT);
3  }
4
5  void loop() {
6    digitalWrite(LED_BUILTIN, HIGH);
7    delay(1000);
8    digitalWrite(LED_BUILTIN, LOW);
9    delay(1000);
10 }
```

Serial

- Gibt `"Hello, World!"` auf der Konsole aus
- Spiegelt anschließend die Texteingabe zurück
- Benutzt `setup()` und `loop()`
- Benutzt `Serial.*`

```
1 void setup() {
2   Serial.begin(19200);
3   Serial.print("Hello, World!\r\n");
4 }
5
6 void loop() {
7   Serial.write(Serial.read());
8 }
```

Pro/Contra Arduino

Pro:

- Man kommt schnell zum Ziel
- Zugrunde liegened Hardware ist schnell ausgetauscht
- Es gibt viele Beispiele für die Arduino-Welt
- „Batterien inklusive“

Contra:

- Es passiert sehr viel *Magie*
- Arduino-Programme sind „fett und träge“
- Wir haben keine wirkliche Kontrolle über das Programm

In vier Schritten zum Ziel

1. Dokumentation lesen
2. Code schreiben
3. Compilen & Linken
4. Flashen

Warum tun wir uns das an?

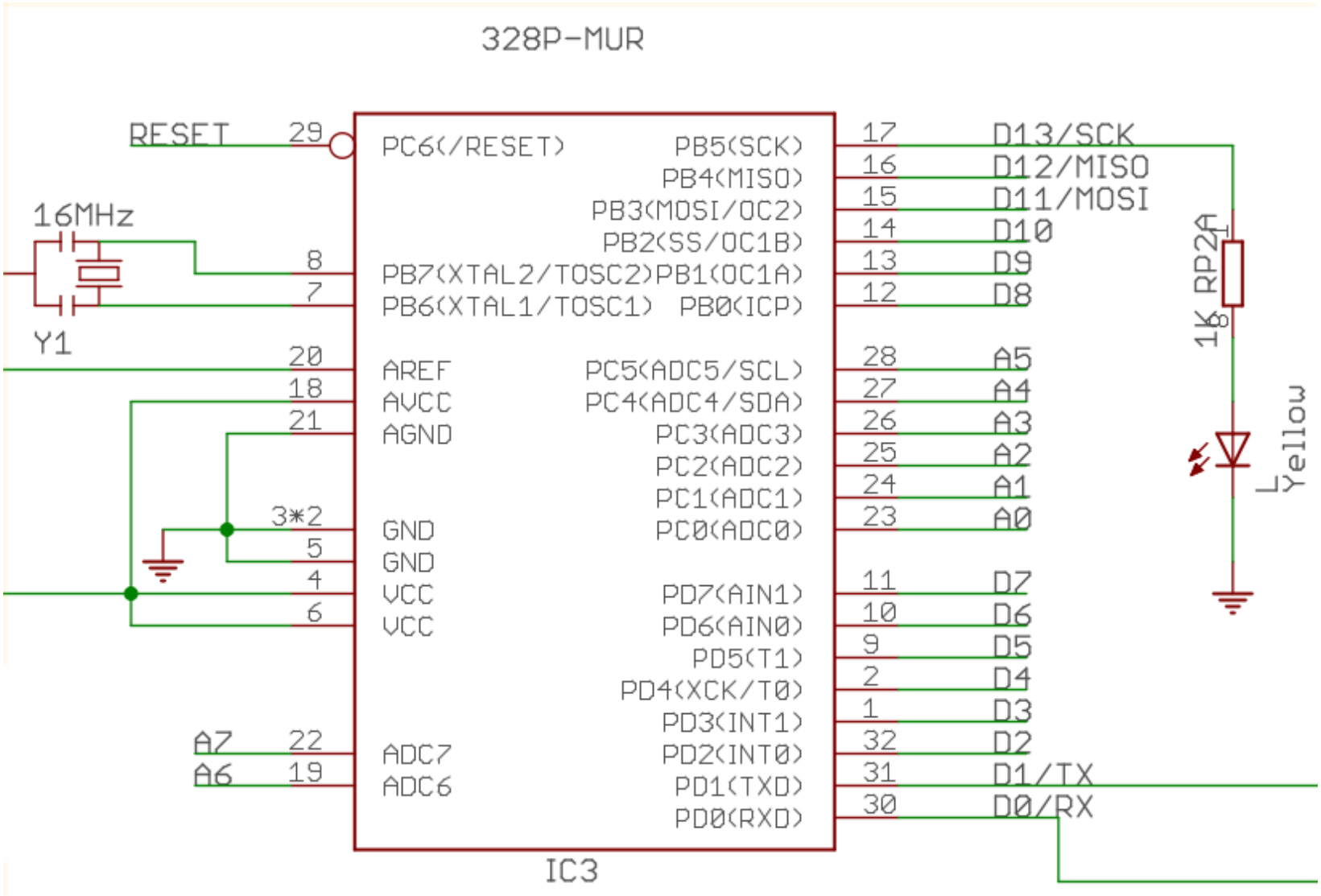
- Programme werden kleiner
- Programme werden schneller
- Wir können die Hardware voll ausnutzen

Schritt 1: Dokumentation lesen

Welche Dokumente brauchen wir?

- Schaltplan
- Datenblatt

Schaltplan


328P-MUR
RESET
29
PC6(/RESET)
PB5(SCK)
PB4(MISO)
PB3(MOSI/OC2)
PB2(SS/OC1B)
PB7(XTAL2/TOSC2)
PB1(OC1A)
PB6(XTAL1/TOSC1)
PB0(ICP)
16MHz
8
7
Y1
20
AREF
18
AVCC
21
AGND
3*2
GND
5
GND
4
VCC
6
VCC
A7
22
ADC7
A6
19
ADC6
PC5(ADC5/SCL)
PC4(ADC4/SDA)
PC3(ADC3)
PC2(ADC2)
PC1(ADC1)
PC0(ADC0)
PD7(AIN1)
PD6(AIN0)
PD5(T1)
PD4(XCK/T0)
PD3(INT1)
PD2(INT0)
PD1(TXD)
PD0(RXD)
17
D13/SCK
16
D12/MISO
15
D11/MOSI
14
D10
13
D9
12
D8
28
A5
27
A4
26
A3
25
A2
24
A1
23
A0
11
D7
10
D6
9
D5
2
D4
1
D3
32
D2
31
D1/TX
30
D0/RX
IC3
1K
RP2A
1
8
Yellow

Schaltplan

- *Ground Truth* für Pinbelegungen
- Zeigt uns, wie die Hardware funktioniert
- Offenbart manchmal undokumentierte Möglichkeiten

Datenblatt

11.3.3 Alternate Functions of Port D

The Port D pins with alternate functions are shown in Table 11-9.

Table 11-9. Port D Pins Alternate Functions

Port Pin	Alternate Function
PD7	AIN1 (Analog Comparator Negative Input) PCINT23 (Pin Change Interrupt 23)
PD6	AIN0 (Analog Comparator Positive Input) OC0A (Timer/Counter0 Output Compare Match A Output) PCINT22 (Pin Change Interrupt 22)
PD5	T1 (Timer/Counter 1 External Counter Input) OC0B (Timer/Counter0 Output Compare Match B Output) PCINT21 (Pin Change Interrupt 21)
PD4	XCK (USART External Clock Input/Output) T0 (Timer/Counter 0 External Counter Input) PCINT20 (Pin Change Interrupt 20)

Datenblatt

- *Ground Truth* für Hardware-Funktionalität
- Zeigt uns, wie der Microcontroller funktioniert
- Hilfreich: Doppelfunktionen für Pin-Belegungen

Schritt 2: Code schreiben

- Blink 1 / How to: Überhaupt - Blink 2 / How to: Datenblatt verwenden - UART 1 / How to: Hardware-Init - UART 2 / How to: Interrupt

How to: Code – Blink 1

How to: Datenblatt – Blink 2

How to: Hardware-Init – UART 1

How to: Interrupts – UART 2

Schritt 3: Compilen & Linken

- Was ist ein - Compiler? - Linker? - Bibliothek? - Makefile? - Dateiformate - Tools

Was ist ein Compiler?

Was ist ein Linker?

Was ist eine Bibliothek?

Was ist ein Makefile?

Dateiformate

- Dateiformate - elf object - elf executable - ihex

Tools

- objdump - size - addr2line

objdump

size

addr2line

Schritt 4: Flashen

- Möglichkeiten - High Voltage Serial Programming - In-System Programming - Bootloader - Tools - avrdude - PonyProg - AVRStudio - ...

In-System Programming

Bootloader

avrdude

Was haben wir gewonnen?

- Wir haben die Kontrolle über den Code
- Wir können die Hardware voll ausnutzen
- Unsere Programme sind kleiner
- Unsere Programme benötigen weniger CPU-Zeit

Vergleich: Blink

Vergleich: Serial

Wie geht's weiter?

- Microcontroller.net - Roboternetz.de - Quellcode aus den Slides

Fragen?