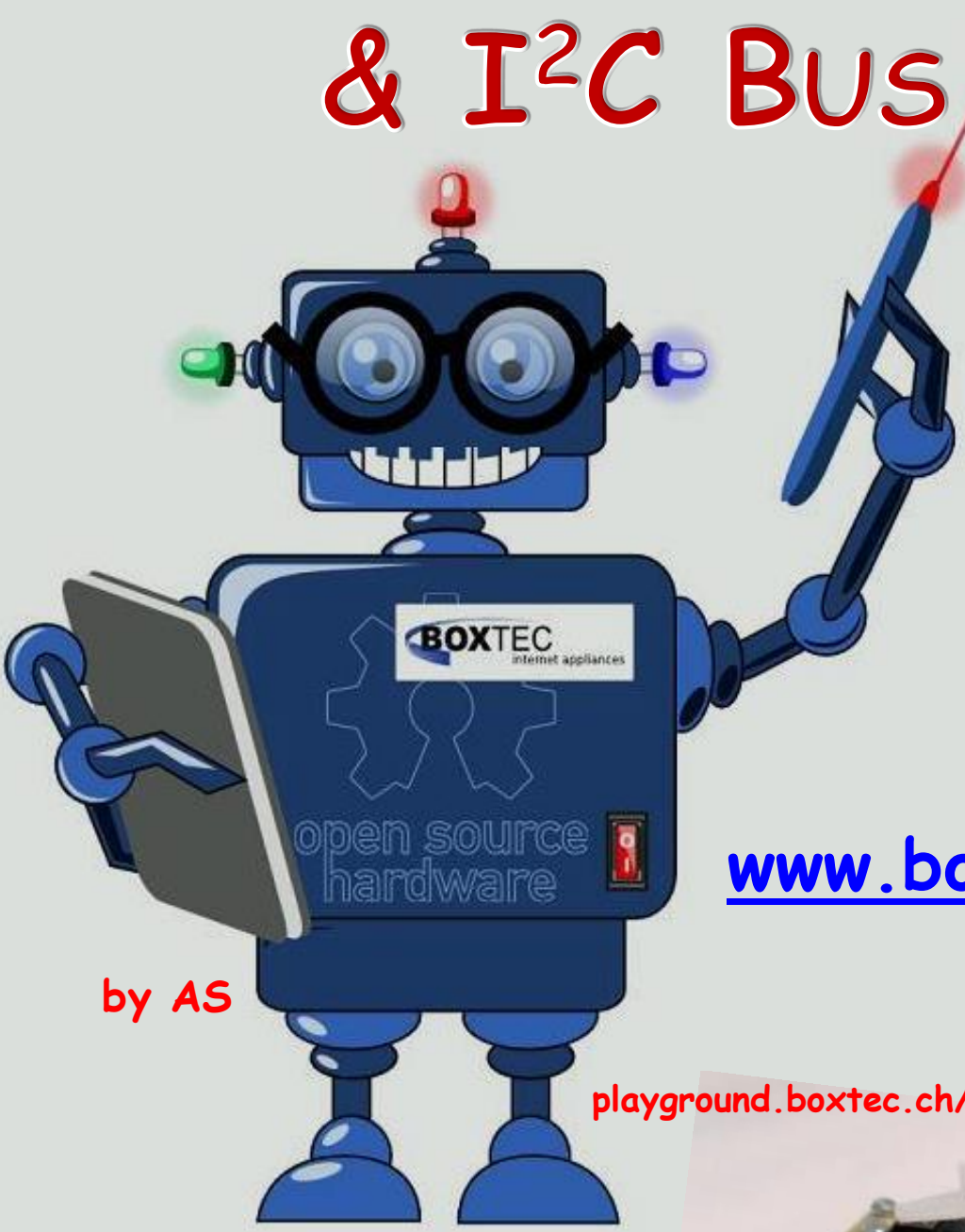
Mikrokontroller & I²C Bus



www.boxtec.ch

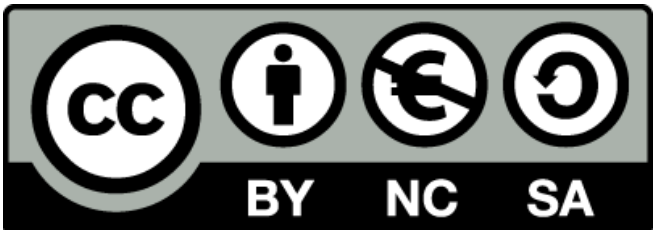
by AS

playground.boxtec.ch/doku.php/tutorial

Attiny 841 – Servo
Teil 3 – Software

I²C Bus und der Attiny 841

Copyright



Sicherheitshinweise

Lesen Sie diese Gebrauchsanleitung, bevor Sie diesen Bausatz in Betrieb nehmen und bewahren Sie diese an einem für alle Benutzer jederzeit zugänglichen Platz auf. Bei Schäden, die durch Nichtbe-achtung dieser Bedienungsanleitung verursacht werden, erlischt die Gewährleistung/Garantie. Für Folgeschäden übernehmen wir keine Haftung! Bei allen Geräten, die zu ihrem Betrieb eine elektrische Spannung benötigen, müssen die gültigen VDE-Vorschriften beachtet werden. Besonders relevant sind für diesen Bausatz die VDE-Richtlinien VDE 0100, VDE 0550/0551, VDE 0700, VDE 0711 und VDE 0860. Bitte beachten Sie auch nachfolgende Sicherheitshinweise:

- Nehmen Sie diesen Bausatz nur dann in Betrieb, wenn er zuvor berührungssicher in ein Gehäuse eingebaut wurde. Erst danach darf dieser an eine Spannungsversorgung angeschlossen werden.
- Lassen Sie Geräte, die mit einer Versorgungsspannung größer als 24 V- betrieben werden, nur durch eine fachkundige Person anschließen.
- In Schulen, Ausbildungseinrichtungen, Hobby- und Selbsthilfewerkstätten ist das Betreiben die-ser Baugruppe durch geschultes Personal verantwortlich zu überwachen.
- In einer Umgebung in der brennbare Gase, Dämpfe oder Stäube vorhanden sind oder vorhanden sein können, darf diese Baugruppe nicht betrieben werden.
- Im Falle eine Reparatur dieser Baugruppe, dürfen nur Original-Ersatzteile verwendet werden! Die Verwendung abweichender Ersatzteile kann zu ernsthaften Sach- und Personenschäden führen. Eine Reparatur des Gerätes darf nur von fachkundigen Personen durchgeführt werden.
- Spannungsführende Teile an dieser Baugruppe dürfen nur dann berührt werden (gilt auch für Werkzeuge, Messinstrumente o.ä.), wenn sichergestellt ist, dass die Baugruppe von der Versor-gungsspannung getrennt wurde und elektrische Ladungen, die in den in der Baugruppe befindlichen Bauteilen gespeichert sind, vorher entladen wurden.
- Sind Messungen bei geöffnetem Gehäuse unumgänglich, muss ein Trenntrafo zur Spannungsver-sorgung verwendet werden
- Spannungsführende Kabel oder Leitungen, mit denen die Baugruppe verbunden ist, müssen immer auf Isolationsfehler oder Bruchstellen kontrolliert werden. Bei einem Fehlers muss das Gerät un-verzüglich ausser Betrieb genommen werden, bis die defekte Leitung ausgewechselt worden ist.
- Es ist auf die genaue Einhaltung der genannten Kenndaten der Baugruppe und der in der Baugrup-pe verwendeten Bauteile zu achten. Gehen diese aus der beiliegenden Beschreibung nicht hervor, so ist eine fachkundige Person hinzuzuziehen

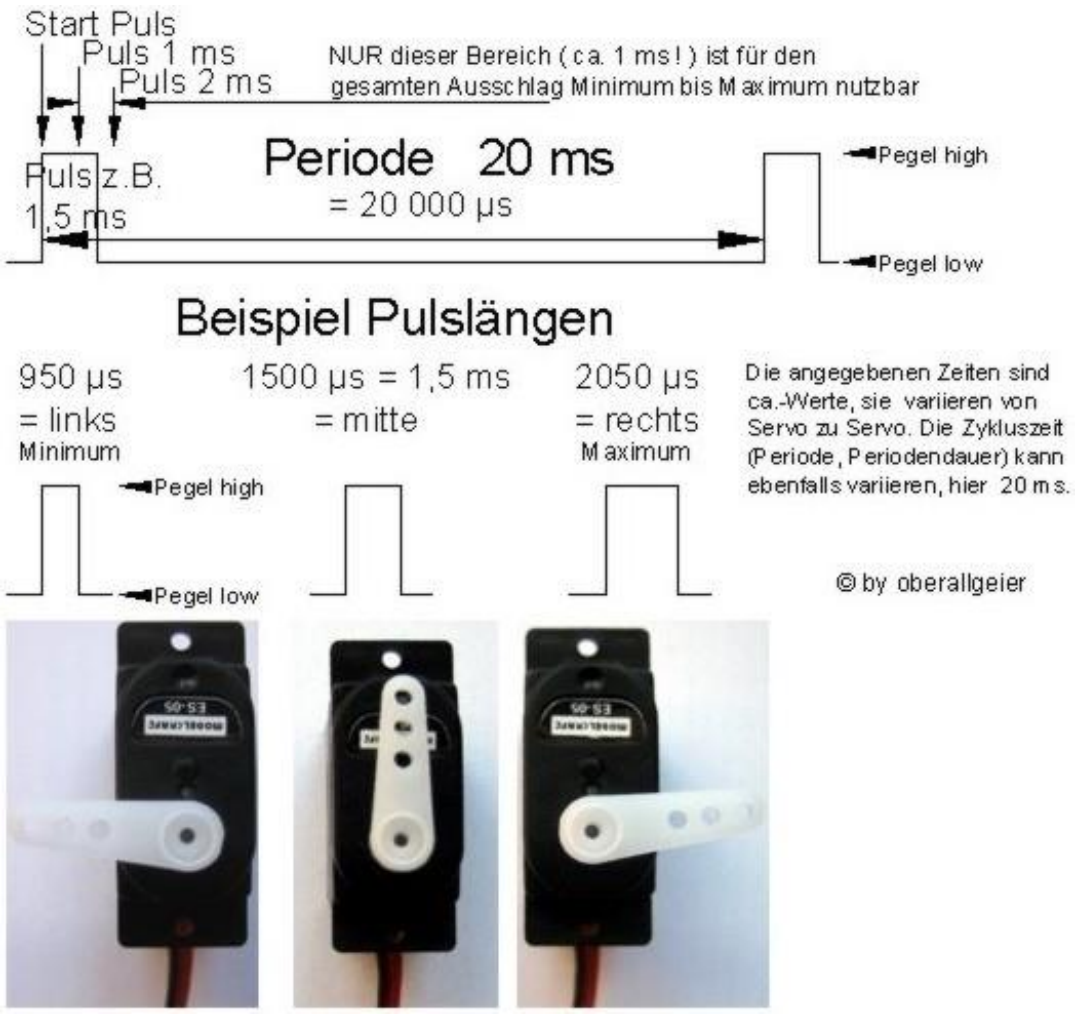
Bestimmungsgemäße Verwendung

- Auf keinen Fall darf 230 V~ Netzspannung angeschlossen werden. Es besteht dann Lebensgefahr!
- Dieser Bausatz ist nur zum Einsatz unter Lern- und Laborbedingungen konzipiert worden. Er ist nicht geeignet, reale Steuerungsaufgaben jeglicher Art zu übernehmen. Ein anderer Einsatz als angegeben ist nicht zulässig!
- Der Bausatz ist nur für den Gebrauch in trockenen und sauberen Räumen bestimmt.
- Wird dieser Bausatz nicht bestimmungsgemäß eingesetzt kann er beschädigt werden, was mit Ge-fahren, wie z.B. Kurzschluss, Brand, elektrischer Schlag etc. verbunden ist. Der Bausatz darf nicht geändert bzw. umgebaut werden!
- Für alle Personen- und Sachschäden, die aus nicht bestimmungsgemäßer Verwendung entstehen, ist nicht der Hersteller, sondern der Betreiber verantwortlich. Bitte beachten Sie, dass Bedien- und /oder Anschlussfehler außerhalb unseres Einflussbereiches liegen. Verständlicherweise können wir für Schäden, die daraus entstehen, keinerlei Haftung übernehmen.
- Der Autor dieses Tutorials übernimmt keine Haftung für Schäden. Die Nutzung der Hard- und Software erfolgt auf eigenes Risiko.

ATtiny 841 - Servo Teil 3 – Software

Sehen wir uns noch mal die Grundlagen zum Servo an:
An die Signalleitung wird ein pulsweitenmoduliertes Signal (PWM) angeschlossen. Die Periode entspricht bei den meisten Modellen 20ms. Innerhalb/zu Beginn dieser 20ms wird ein Puls erwartet, der sich zwischen 1ms und 2ms bewegt, wobei diese Werte jeweils den Endlagen des Servos entsprechen. D.h. 1ms ist ganz links und 2ms ist ganz rechts (Einige Servos haben in diesem Wertebereich jedoch nicht die volle Bewegungsfreiheit ausgenutzt, die Werte, bei denen der Servo ganz links/rechts ist können auch unter 1ms/über 2ms liegen). 1,5ms würde demnach die Mittelstellung bedeuten. Aufgrund der Pulslänge lässt sich also eine direkte Aussage über die Position des Servos treffen. Der Motor sorgt dann intern mithilfe des Potis dafür, dass die Position gehalten wird.

Sehen wir uns noch mal das Impulsdiagramm an:



In Abhängigkeit einer Impulslänge wird die Position eines Servos gezeigt. Dabei entstehen 3 Stellungen:

- 950 µs – Stellung links
- 1500 µs – Stellung mitte
- 2050 µs – Stellung rechts

Mit diesem Wissen können wir unser erstes einfaches Programm schreiben:

```
/* ATB_Ati841_P175_Servo_Prg1.c*/
// Programm für Attiny 841 mit Quarz mit Platine 175 Uni - Servo
// Platine P175 Attiny 841 mit Quarz
// PA0 --> LED 2
// PA1 --> LED 3
// PA2 --> LED 4
// PA3 --> LED 5
// PA5 --> Servo PWM
// PA7 --> US Trigger
// PB2 --> US Echo
// Fuse Einstellung mit externem Quarz
// Ex - 0xFF
// Hi - 0xDF
// Lo - 0xCF

#define F_CPU 16000000UL                // Angabe der Frequenz, wichtig für die Zeit
#include "util/delay.h"                 // Einbindung Datei Pause
#include "avr/io.h"                     // Einbindung Datei Ausgänge
int main(void)
 {
   DDRA=0b00100000;                     // DDR A auf Ausgang schalten
   PORTA |= (1<<PINA5);                 // Ausgang einschalten
   while(1)                             // Programmschleife
    {
      while( 1 )                        // Wiederholung
       {
         _delay_us(1500);               // Zeit Drehung
                                        // Begrenzung beachten – von ca. 1000us bis 2000us
         PORTA &= ~(1<<PINA5);          // Ausgang abschalten
          _delay_ms(20);                // Wiederholt alle 20ms
       }
      return 0;
    }
 }
```

Mit der Einstellung _delay_us(1500); wird eine mittlere Stellung des Servos vorgenommen.
Um die Einstellung zu ändern kann man einfach andere Zeiten eingeben. Dabei ist die Begrenzung zu beachten. Am Ende des jeweiligen Bereiches kann es zu einem „Zittern" des Servos kommen oder es erfolgt eine Blockade. Im Kommentar habe ich den Bereich angegeben in dem mein Servo funktionierte. Das kann allerdings schwanken.

Das folgende Programm habe ich mit dem Mini Board 1 getestet. Mit den Taster 1, 2 und 3 kann ich verschiedene Positionen Anfahren. Diese Positionen lassen sich im Programm einstellen. Der Bereich muss getestet werden, da sie bei jedem Servo unterschiedlich sein können.

```
// Programm für Attiny 841 mit Quarz mit Platine 175
// Uni - Servo
// PA1 --> Taster 3
// PA2 --> Taster 2
// PA3 --> Taster 1
// PA5 --> Servo PWM
// nicht belegt
// nicht belegt
// Fuse Einstellung mit externem Quarz
// Ex - 0xFF
// Hi - 0xDF
// Lo - 0xCF

#define F_CPU 16000000UL
#include <avr/io.h>
#include <util/delay.h>

void waitus (uint16_t us)                  // Unterprogramm Pause
 {
   for(; us>0; us--)_delay_us(1);
 }

#define taste1 (PINA & (1<<PINA1))         // Taste 1
#define taste2 (PINA & (1<<PINA2))         // Taste 2
#define taste3 (PINA & (1<<PINA3))         // Taste 3

uint16_t stellung = 1100;                  // Stellung beim einschalten
uint16_t stellungmitt = 1500;              // Stellung Mitte Taste 2
uint16_t stellungmax = 1800;               // Stellung Maximal Taste 1
uint16_t stellungmin = 650;                // Stellung Minimal Taste 3

int main (void)
 {
   DDRA=0b00100000;                              // DDR A auf Ausgang schalten
    while( 1 )
     {
       if (!(taste1))                      // Abfrage Taste 1
        {
          _delay_ms(20);                 // Tasterentprellung
          if (!(taste1))                 // Abfrage Taste 1
            {
              stellung = stellungmax;
            }
        }
       if (!(taste2))                      // Abfrage Taste 2
        {
          _delay_ms(20);                 // Tasterentprellung
          if (!(taste2))                 // Abfrage Taste 2
            {
              stellung = stellungmitt;
```

```
            }
          }
        if (!(taste3))                    // Abfrage Taste 3
          {
            _delay_ms(20);                // Tasterentprellung
            if (!(taste3))                // Abfrage Taste 3
              {
                stellung = stellungmin;
              }
          }
        PORTA |= (1<<PINA5);              // setzt Servo - ein
        waitus(stellung);                 // Aufruf Unterprogramm
        PORTA &= ~(1<<PINA5);             // setzt Servo - aus
        _delay_ms(20);                 // ist nicht kritisch
      }
    return 0;
}
```

Einige Teile des Textes wurden zur besseren Übersicht farblich gestaltet.
Die Nutzung erfolgt auf eigenes Risiko.

Ich wünsche viel Spaß beim Bauen und programmieren
Achim

Quellenangabe:
https://www.mikrocontroller.net/articles/AVR-Tutorial:_Servo
https://www.mikrocontroller.net/articles/Modellbauservo_Ansteuerung

verschiedene Artikel aus https://www.mikrocontroller.net und
https://rn-wissen.de/wiki/index.php/Hauptseite

Programmen von Peter Dannegger, Stefan Frinks und oberallgeier