

TU Berlin Fakultät IV
Institut für Energie und Automatisierungstechnik
Fachgebiet Elektronische Mess- und Diagnosetechnik
Praktikum Messdatenverarbeitung

Praktikum Messdatenverarbeitung

Termin 1

Özgü Dogan (326 048)
Timo Lausen (325 411)
Boris Henckell (325 779)

3. Mai 2012

Gruppe: G1 Fr 08-10

Betreuer: Jürgen Funk

Inhaltsverzeichnis

1	Vorbereitungsaufgaben	1
1.1	Quellcode	1
1.2	Abtastrate des ADU	2
2	Versuch	2
3	Ergebnisse	2

1 Vorbereitungsaufgaben

1.1 Quellcode

```
// Vorbereitungsaufgabe Termin 2
//      zg      Dogan (326048)
// Timo Lausen (325411)
// Boris Henckell (325779)

#include <avr/io.h>
#include <avr/interrupt.h>
#include <inttypes.h>

ISR(ADC_vect)
{
    uint16_t ADUWERT = ADC;
    PORTC &= ~(1<<PC5);    // anschalten der Roten LED
    if (ADUWERT<=340){
        PORTC |= (1<<PC1); // Orangene LED ausgeschaltet
        PORTC |= (1<<PC4); // Grüne LED ausgeschaltet
    }
    else if (ADUWERT<=682){
        PORTC |= (1<<PC1); // Orangene LED ausgeschaltet
        PORTC &= ~(1<<PC4); // Grüne LED anschalten
    }
    else {
        PORTC &= ~(1<<PC1); // Orangene LED anschalten
        PORTC &= ~(1<<PC4); // Grüne LED anschalten
    }
}

int
main (void)
{
    // ADU anschalten

    ADCSRA |= (1<<ADEN);    // shiften 1 in ADCSRA um ADEN, damit ADU angeht

    // ADCH und ADCL Register (je 8bit)
    // 10bit rechts auslesen – ganz normal
    // links adjusten
    // ADMUX |= (1<<ADLAR);    // shiften 1 in ADMUX um ADLAR

    ADMUX |= (1<<REFS1) | (1<<REFS0);    // Voltage reference 2.56V (s. Datenblatt)
    ADCSRA |= (1<<ADPS0) | (1<<ADPS2);    // F_adu = (F_clk)/32 (s. Datenblatt)

    // Als Eingang soll der Kanal ADC0 im Single-Ended-Modus genutzt werden.
    // Dafür wird im ADCSRB Register MUX 4:0 zu null gesetzt. Der 5. Bit (MUX5)
    // (s. Datenblatt S.290, 26.8.2)

    // Für den free running modus müssen die ADTS2,1,0 Bits im ADCSRB Register
    // zu null gesetzt werden.
```

```
// ADCSRB &= ~(1<<ADTS2) & ~(1<<ADTS1) & ~(1<<ADTS0);  
(s. Datenblatt S.296, Tabelle 26–6)  
  
sei(); // aktiviert Interrupts allgemein  
  
// alternativ Interrupts global aktivieren: //(s. Datenblatt S.14, 7.4.1)  
// SREG |= (1<<I);  
  
ADCSRA |= (1<<ADIE); // aktiviert das Interrupt im ADC  
  
while ( 1 ) {  
}  
return 0 ;  
}
```

1.2 Abtastrate des ADU

Der ADU arbeitet bei dieser Einstellung mit einer Abtastrate von $\frac{f_{clk}}{32}$. Es lassen sich auch Abtastraten von $\frac{f_{clk}}{2}$, $\frac{f_{clk}}{4}$, $\frac{f_{clk}}{8}$, $\frac{f_{clk}}{16}$, $\frac{f_{clk}}{64}$ und $\frac{f_{clk}}{128}$ einstellen.

Für eine Umsetzung benötigt der ADU-jedoch 13 solcher Takte und daher ist die effective Abtstrate $f_{sample} = \frac{f_{clk}}{13 \cdot 32}$.

Im Datenblatt ist angegeben, dass ein ADU, der eine kleinere Auflösung als 10 Bit benötigt (in unserem Fall arbeiten wir mit 8 Bit) mit einer Clock arbeiten, der eine Frequenz zwischen 50 und 200kHz hat. Wenn wir von 200kHz ausgehen würde die eingestellte Abtastrate mit $\frac{200kHz}{13 \cdot 32} \approx 480Hz$ betragen.

Die Clock-Frequenz kann auch bis zu 1MHz hochgehen, falls eine höhere Abtastrate erforderlich ist.

2 Versuch

3 Ergebnisse