**PROTOKOLL**zur Laborübung

***AVR ext. Interrupt, Timer und PWM***



|  |  |  |
| --- | --- | --- |
| Gruppe / Klasse | Protokollführer | Unterschrift |
| 5 / **4BHELS** | HOFSTÄTTER A. |  |
| Übungs- / Abgabedatum | Mitarbeiter | Unterschrift |
| 4. Nov. 2014  11. Nov. 2014 | HIRSCH L. |  |
| Lehrer | Mitarbeiter | Unterschrift |
| CRHA |  |  |
| Note | Mitarbeiter | Unterschrift |
|  |  |  |
| ***AVR ext. Interrupt, Timer und PWM***  *ATmega32U4* | | |
| **Verwendete Geräte**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | Nr. | Gerätebezeichnung | Hersteller | Typ | Platznummer | | 1. | Oszilloskop | Tektronix | TDS 1001B | - |   **Verwendete Programme**   |  |  |  | | --- | --- | --- | | Nr. | Name | Version | | 1. | CodeBlocks | 13.12 | | 2. | DFU-Programmer | 1.2.2 | | | |

ÜBUNGS-/ABGABE-DATUM

Klasse /Gruppe

NOTE

LEHRER

# Inhaltsverzeichnis

[1 Inhaltsverzeichnis 2](#_Toc403409514)

[2 Aufgabenstellung 3](#_Toc403409515)

[2.1 Pin Change Interrupt 3](#_Toc403409516)

[2.2 Counter 0 3](#_Toc403409517)

[2.3 Timer 0 Compare 3](#_Toc403409518)

[3 Led toggeln (Pin Change Interrupt) 4](#_Toc403409519)

[3.1 Schaltung 4](#_Toc403409520)

[3.2 Programmlisting 4](#_Toc403409521)

[4 Zählen von Flanken – 3 Bit (Counter 0 ) 5](#_Toc403409522)

[4.1 Schaltung 5](#_Toc403409523)

[4.2 Programmlisting 5](#_Toc403409524)

[4.3 Alternatives Programmlisting (ohne Counter) 5](#_Toc403409525)

[5 Timer 0 Compare 6](#_Toc403409526)

[5.1 Programmlisting 6](#_Toc403409527)

[5.2 Messergebnisse 6](#_Toc403409528)

[6 Allgemeines Kommentar 7](#_Toc403409529)

[7 Abbildungsverzeichnis 8](#_Toc403409530)

[8 Programmlistingverzeichnis 8](#_Toc403409531)

# Aufgabenstellung

Folgende 3 Aufgabenstellungen waren zu erledigen.

## Pin Change Interrupt

„Toggle eine LED am Pin PD0 bei jeder Pegeländerung an den Pins ***PCINT7:0***. Dies soll mit Hilfe des Pin Change Interrupts durchgeführt werden.”

## Counter 0

„Zähle mit Hilfe des Counter 0 die negativen Flanken am Pin ***T0*** (PD7) und gib die Anzahl binär an 3 LEDs

aus. Der maximale Zählerstand bei 3 Bit () war zu berücksichtigen.“

## Timer 0 Compare

„Erzeuge ein Rechtecksignal mit Hilfe von ***Timer 0 Compare*** am Pin PB0 mit folgenden Eckdaten:“

# Led toggeln (Pin Change Interrupt)

Es sollte der Pin PB0 bei jeder Pegeländerung an dem Pin PCINT7 (PB7) mit Hilfe des Pin Change Interrupt getoggelt werden. Um dies zu überprüfe wurde der PIN PB0 mit einer LED beschalten.

## Schaltung

Um einen fehlerfreien Betrieb zu garantieren musste für den Trigger Pin einer der Ports zwischen PCINT0 bis PCINT7 gewählt werden.

Um eine visuelle Trennung zwischen Ein- und Ausgang zu gewährleisten wurde für den Eingang der höchste PCINT Pin 7 gewählt. Dieser beläuft sich auf den Namen PB7 (bzw. PCINT7).

Wäre der Taster Eingang nicht auf einen dieser Pin Change Interrupt Pins würde die Aufgabenstellung logischerweise nicht funktionieren, da somit kein Triggern im Rahmen des Interrupts erfolgen würde.

Abbildung 1 – Schaltung Aufgabe 1

## Programmlisting

1 **|** #include **<**avr**/**io**.**h**>** // Include File für IO Definitionen

2 **|** #include **<**avr**/**interrupt**.**h**>** // Include File für sei(), cli()

3 **|**

4 **|** int main**(**void**)**

5 **|** **{**

6 **|** DDRB **|=** **(**1**<<**DDB0**);** // PB0 als Ausgang

7 **|** DDRB **&=~** **(**1**<<**DDB7**);** // PB7 als Eingang

8 **|**

9 **|** PORTB **|=** **(**1**<<**PB7**);** // Internen Pull Up einschalten

10 **|**

11 **|** PCICR**|=(**1**<<**PCIE0**);**

12 **|** PCMSK0**|=(**1**<<**PCINT7**);** // Interrupt global freigeben

13 **|** sei**();** // Global freigeben

14 **|** **while(**1**);** // Programmschelife

15 **|** **}**

16 **|** ISR**(**PCINT0\_vect**)**

17 **|** **{**

18 **|** PORTB**^=(**1**<<**PB0**);** // Ausgang PB0 Toggeln

19 **|** **}**

**Listing 1** – C Code für Pin Change Interrupt (Aufgabe 1)

# Zählen von Flanken – 3 Bit (Counter 0 )

Die Aufgabe bestand darin, mit Hilfe des Counters 0 die negativen Flanken am T0 Pin zu zählen und die Anzahl mit 3 LEDs anzuzeigen. Dabei muss berücksichtigt werden, dass maximal die Zahl 7 ausgegeben werden kann.

## Schaltung

Um die 3 Bit Codierung visuell darzustellen wurden 3 Leds verwendet.

Nach betätigen des Tasters am Pin PD7 wird TCNT0 solange hochgezählt und dem kompletten Port B zugewiesen bis dieser beim maximalen Zählerstand von 7 (OCR0A) im Interrupt zurückgesetzt wird.

Der Taster an Port PD7 darf keinesfalls über einen Treiber IC betrieben werden da es hier zu einem erheblichen Fehlverhalten kommen kann

Abbildung 2 – Schaltung Aufgabe 2

## Programmlisting

1 **|** #include **<**avr**/**io**.**h**>** // Include File für IO Definitionen

2 **|** #include **<**avr**/**interrupt**.**h**>** // Include File für sei(), cli()

3 **|**

4 **|** int main**(**void**)**

5 **|** **{**

6 **|** DDRB **=** 0xFF**;** // PB7...PB0 als Ausgang

7 **|** PORTD **|=** **(**1**<<**PD7**);** // PD7 als Eingang

8 **|**

9 **|** TCCR0B**|=(**1**<<**CS02**)|(**1**<<**CS01**);** // Clock bei fallender Flanke

10 **|** TIMSK0**|=(**1**<<**OCIE0A**);** // Interrupt freigeben

11 **|**

12 **|** OCR0A**=**7**;** // maximaler Zählerstand 7 (0 bis 7)

13 **|** sei**();** // global freigeben

14 **|** **while(**1**)**

15 **|** **{** PORTB**=**TCNT0**;** // Zuweisung des Timer/Counter Register

16 **|** **}**

17 **|** **}**

18 **|** ISR**(**TIMER0\_COMPA\_vect**)**

19 **|** **{**

20 **|** TCNT0**=**0**;** // Reset; Zählerstand auf 0 zurücksetzen

21 **|** **}**

**Listing 2** – C Code für Implementierung von Counter 0 (Aufgabe 2)

## Alternatives Programmlisting (ohne Counter)

Zu Demonstrationszwecken wurde oben ausgeführtes Programm zusätzlich auch ohne den Gebrauch von Counter0 ausprogrammiert.

1 **|** int main**(**void**)**

2 **|** **{**

3 **|** DDRB **=** 0xFF**;** //PB7...PB0 als Ausgang

4 **|** DDRD **=** DDRD**&~(**1**<<**DDD2**)&~(**1**<<**DDD0**);** //DDRD6,0 = 0 (Input)

5 **|** PORTD **=** PORTD**|(**1**<<**PD2**)|(**1**<<**PD0**);** //PORTD6,0= 1 (Pull Up on)

6 **|**

7 **|** EICRA **=** EICRA **|** **(**1**<<**ISC01**);** //INT0 fallende Flanke

8 **|** EIMSK **=** EIMSK **|** **(**1**<<**INT0**);** //INT0 freigeben

9 **|** sei**();** //Interrupts generell freigeben

10 **|**

11 **|** **while(**1**)**

12 **|** **{** **while** **(**PIND **&** **(**1**<<**PD3**));** //warten auf fallende Flanke

13 **|** PORTB **=** 0x00**;** //Port B löschen

14 **|** **while** **(!(**PIND **&** **(**1**<<**PD3**)));** //warten auf steigende Flanke

15 **|** **}**

16 **|** **return** 0**;**

17 **|** **}**

18 **|** ISR**(**INT0\_vect**)** //Interruptroutine INT0

19 **|** **{** PORTB**++;**

20 **|** **if** **(**PORTB **==** 7**)** //Bei Grenzwert (Dez=7)

21 **|** PORTB **=** 0**;**  // Port B löschen

22 **|** **}**

**Listing 3** – C Code für alternative Implementierung der Aufgabe 2 (ohne Counter)

# Timer 0 Compare

Die Aufgabe bestand darin, ein Rechteck Signal am PB0 mit tEIN = tAUS = 500ms zu erzeugen. Dies soll mit Hilfe des Timer 0 Compare gelöst werden. Anschließend ist es mit dem Oszilloskop zu dokumentieren.

Um dies zu realisieren, wurde der folgende Code verwendet:

## Programmlisting

1 **|** #include **<**avr**/**io**.**h**>** // Include File für IO Definitionen

2 **|** #include **<**avr**/**interrupt**.**h**>** // Include File für sei(), cli()

3 **|**

4 **|** int tEIN **=** 500**;** // Globale Variable von tEIN in ms

5 **|**

6 **|** ISR **(**TIMER0\_COMPA\_vect**)** // Timer 0 Interrupt

7 **|** **{** static int anzahl **=** 0**;** // Speicher Wert bei Wiederaufruf

8 **|** anzahl**++;** // Bei jeden Aufruf erhöhen

9 **|** **if** **(**anzahl **==** tEIN**/**8**)**

10 **|** **{** PORTB **^=** **(**1**<<**PORTB0**);** // Toggeln von PB0

11 **|** TCNT0 **=** 0**;** // Reset Timer/Counter0 Compare Register

12 **|** anzahl **=** 0**;** // Reset Anzahl der Aufrufe

13 **|** **}**

14 **|** **}**

15 **|** int main **(**void**)**

16 **|** **{**

17 **|** DDRB **=** DDRB **|** **(**1**<<**DDB0**);** // PB0 als Ausgang

18 **|**

19 **|** TIMSK0 **=** TIMSK0**|(**1**<<**OCIE0A**);**

20 **|** OCR0A **=** 30**;** // Endwert für Aufruf jede ms

21 **|** TCCR0B **|=** **(**1**<<**CS01**)|(**1**<<**CS00**);** // Teiler 64 => dt=64/16MHz=4us

22 **|** sei**();** // Interrupt global freigeben

23 **|** **while(**1**);**

24 **|** **}**

**Listing 4** – C Code für die Implementierung eines Timer 0 Compare Interrupt (Aufgabe 3)

## Messergebnisse

Um die Korrektheit des Rechtecksignals am Ausgangspin PB0 zu prüfen wurde mit einem Oszilloskope eine Messung zwischen PB0 und Masse durchgeführt.

Aufgrund der gewählten Zeit pro Division von 250ms sind tEIN bzw. tAUS in der Höhe von jeweils 500ms sehr gut erkennbar. Diese Genauigkeit ergibt sich durch die korrekte Berechnung des Registers OCR0A und der kalkulierten Aufrufverzögerung des Interrupts wie im Listing 4 (Zeile 9 und 20) erkennbar ist.



Abbildung 3 – Ausgangssignal am Pin PB0 (Rechteck T=1s)

# Allgemeines Kommentar

Da bei diversen Peripheriebeschaltungen (Zusatzboards) oftmals Treiber ICs für die Ansteuerung von Port Pins genutzt werden muss darauf geachtet werden, dass alle Port Pins direkt vom ATmega32U4 abgegriffen werden.

Andernfalls werden z.B. externer Pin Interrupts nicht an Schaltern mit Treiber ICs nicht (richtig) getriggert.

# Abbildungsverzeichnis

[Abbildung 1 – Schaltung Aufgabe 1 4](file:///C:\Users\Alex\Dropbox\Schule\4-Jahrgang\Labor\CRHA\1_AVR_ext_Interrupt_Timer_PWM\4BHELS_5_CRHA_Hofstaetter_AVR_ext_Interrupt_Timer_PWM.docx#_Toc403409532)

[Abbildung 2 – Schaltung Aufgabe 2 5](file:///C:\Users\Alex\Dropbox\Schule\4-Jahrgang\Labor\CRHA\1_AVR_ext_Interrupt_Timer_PWM\4BHELS_5_CRHA_Hofstaetter_AVR_ext_Interrupt_Timer_PWM.docx#_Toc403409533)

[Abbildung 3 – Ausgangssignal am Pin PB0 (Rechteck T=1s) 7](#_Toc403409534)

# Programmlistingverzeichnis

[Listing 1 – C Code für Pin Change Interrupt (Aufgabe 1) 4](#_Toc403407044)

[Listing 2 – C Code für Implementierung von Counter 0 (Aufgabe 2) 5](#_Toc403407045)

[Listing 3 – C Code für alternative Implementierung der Aufgabe 2 (ohne Counter) 6](#_Toc403407046)

[Listing 4 – C Code für die Implementierung eines Timer 0 Compare Interrupt (Aufgabe 3) 6](#_Toc403407047)