```
/*
 * hase_V2.asm
 *
 *  Created                 : 14.08.2011 18:54:07
 *  Remembered              : 13.12.2011
 *  Vorläufig abgeschlossen : 14.12.2011
 *
 *
 *  Author: Jörg Schröder
 */
;
; Hase V.2.2 final
;
; Diesmal mit Atmel AVRMEGA8
;
; hase_V2.asm
;
;
.NOLIST
.DEVICE ATmega8
.INCLUDE "m8def.inc"
.LIST

; Wir gönnen uns ein, zwei Variable

.DEF ee_wert   = R15
.DEF zaehl1    = R16
.DEF zaehl2    = R17
.DEF zaehl3    = R18
.DEF zaehl4    = R19

.DEF zaehl_ee  = R20
.DEF fixfoxy   = R21

.DEF temp      = R22
.DEF temp2     = R23
.DEF extra     = R24
.DEF engine    = R25
.DEF countdown = R26

; da R26 schon in der Include definiert war, gibt der Assembler ein Warning aus....

; Der Reset Sprungbefehl

RJMP main

; Und hier gehts los
; Erstmal den Stackpointer initialisieren

main:

          ;für Processoren mit 8 Bit Adressraum
          ;LDI zaehl1, RAMEND
          ;OUT SPL,zaehl1

          ldi zaehl1, HIGH(RAMEND)   ; HIGH-Byte der obersten RAM-Adresse
          out SPH, zaehl1
          ldi zaehl1, LOW(RAMEND)    ; LOW-Byte der obersten RAM-Adresse
          out SPL, zaehl1

; Nu Port D auf Eingang schalten

        LDI zaehl1, 0x00
        OUT DDRD, zaehl1

; PullUps aktivieren

        LDI zaehl1, 0xFF
        OUT PORTD, zaehl1
/*
 * Die Pins PD1, PD2, PD3 und PD4 werden per Jumper auf LOW gezogen
 * Durch die internen PullUps sind sie offen = HIGH !
 * Also auch hier die Invertierung beachten!!!
```

```
 * Bedeutung der Jumper:
 * Alles offen (PD0, PD5, PD6 und PD7 sind immer HIGH)
 * Idee: alle Jumper offen = keine Zeitverlängerung = einmaliger Schleifendurchlauf
 * Zeit in Sekunden wird aus dem EEPROM ausgelesen
 * Diese Hauptschleife wird bei offenen Pins genau einmal abgearbeitet
 * Die vier Pins bekommen die Bedeutung 1,2,4 und 8 und bezeichnen die zusätzlichen (!)
 * Durchläufe der Hauptschleife, somit 1 extra bis 15 extra (2 bis 16 Durchläufe)

*/
; Port B wird Ausgang

        LDI engine, 0x01
        OUT DDRB, engine
        LDI engine, 0x00
        OUT PORTB, engine

; Port B0 ist LOW, Motor aus!

; Nun PD einlesen, wichtig sind Bits 4,3,2 und 1
; die Wertigkeit ist 2,4,8 und 16 deshalb erfolgt eine Bearbeitung des Wertes:
; einmal nach rechts schieben um die Wertigkeit anzupassen (das High-Bit PD0 geht dabei
 hopps)
; danach werden die vier lower Bits maskiert, der Wert der anderen Bits ist unerheblich
; In Variable extra steht nun der invertierte Wert (15 bei offenen Pins)
; Ein flottes EOR mit 0x0F dreht die Zahlen um und jetzt stimmt es

        IN extra, PORTD
        ROR extra
        ANDI extra, 0x0F
        LDI zaehl1, 0x0F
        EOR extra, zaehl1

; Um die Hauptschleife einfach zu halten (der Extra Wert gibt ja nur die zusätzlich Dur
chläufe an)
; wird die Variable um eins erhöht

        INC extra

; So, in extra steht nun der Wert der Duchläufe der Hauptschleife drin. Dieser Wert wir
d nur beim Start gelesen, denn
; die Hauptschleife beginnt hier.....

; Nun brauchen wir noch einen Zähler für das EEPROM

        LDI zaehl_ee, 0

; Diese Variable wird hochgezählt und sollte bei 0xFF auf 0 überspringen....

; und noch ein Register mit 0xFF für den Toggle des Motors

        LDI fixfoxy, 0xff

; und ein Countdown, bevor der Spass losgeht
; die Wartezeit nach dem Aktivieren steht in countdown, also 0 bis 256 Sekunden
; für einen Test 30 Sekunden, im Feld FF

        LDI countdown, 0xff
cntdwn:
        rcall eine_sekunde
        DEC countdown
        BRNE cntdwn

; So: jetzt geht es aber wirklich los....

hauptschleife:

; Daten aus dem eeprom holen
        ldi   ZL,low(daten)
        ADD   ZL, zaehl_ee
        ldi   ZH,high(daten)
        rcall EEPROM_read

; in zaehl1 steht der Wert aus dem EEPROM
```

```
; in zaehl1 findet sich jetzt die erste Zeit, beim ersten Lauf die AN-Zeit, beim zweite
n die AUS-Zeit usw.

; Toggeln Sie den Motor JETZT:

        eor engine, fixfoxy
        andi engine, 0x01
        out PORTB, engine

; das sollte gehen, solange engine richtig gesetzt ist...

; hier beginnt die extra-Runde

        mov temp2, extra

; ab hier ohne die Extra-Runde, die wird später außen rum gebaut....

loopmextra:
        mov temp, ee_wert

loopoextra:
        rcall eine_sekunde
        dec temp

; eine Sekunde abgearbeitet
; Sind wir schon fertsch?

        brne loopoextra

; einmal sind wir durch, dann reduzieren wir jetzt die Extra-Runde um eins

        dec temp2
        brne loopmextra

; na, ob das funktioniert?


;       Nächsten Wert aus dem EEPROM abholen

        INC zaehl_ee

; Der Überlauf von FF nach 0 geht vollautomatisch :-)

; forever.....
        RJMP hauptschleife


; ein paar Subroutinen.......


eine_sekunde:
; Zähler-Variablen löschen

        LDI zaehl2,0xf2
        LDI zaehl3,0x00
        LDI zaehl4,0x00

; Ganz äußere Schleife

ganz: INC zaehl2

; äußere Schleife

aeusser: INC zaehl3

; innere Schleife, inner wird angehüpft bis der INC Null ergibt
; also beim 256.mal wird die Schleife verlassen

inner:  NOP
        INC zaehl4
        BRNE inner
        TST zaehl3
        BRNE aeusser
```

```
; Wenn zaehl3 einmal rum ist, dann wird die Schleife verlassen
; andernfalls ab zur oberen Marke .....
; Nun kommen wir auf den ganz äußeren Kern zurück

        TST zaehl2
        BRNE ganz

; Nun verlassen wir das Unterprogramm

        RET


EEPROM_read:
    sbic     EECR,EEWE       ; prüfe ob der vorherige Schreibzugriff
                             ; beendet ist
    rjmp     EEPROM_read     ; nein, nochmal prüfen
    out      EEARH, ZH       ; Adresse laden
    out      EEARL, ZL
    sbi      EECR, EERE      ; Lesevorgang aktivieren
    in       ee_wert, EEDR       ; Daten in CPU Register kopieren
    ret

; Daten im EEPROM definieren

.eseg
daten:
    ; Jedes Pärchen bedeutet Laufzeit/Pausenzeit
    ; Wir haben 8 Pärchen pro Zeile
    ; und 16 Zeilen
    ; macht 128 Pärchen, damit ist das EEPROM halb voll (256Bytes von 512)
    .db 28,14,28,11,27,19,28,28,22,28,18,17,26,26,25,23
    .db 20,10,14,18,28,17,6,12,24,14,10,9,9,17,11,7
    .db 26,10,14,24,24,13,23,16,11,11,29,7,7,25,26,23
    .db 5,11,12,29,24,13,11,18,23,16,23,27,29,5,5,25
    .db 10,14,20,29,23,13,16,29,20,16,7,23,11,29,17,12
    .db 11,24,12,5,8,18,19,26,5,12,24,29,12,24,25,17
    .db 9,15,17,27,24,29,27,15,15,5,8,21,29,20,29,11
    .db 15,11,11,18,25,26,15,25,9,9,24,16,29,19,29,8
    .db 5,17,6,24,16,28,10,26,28,13,18,28,29,18,9,14
    .db 24,16,28,19,12,13,15,16,17,9,28,16,23,28,20,24
    .db 15,21,18,27,19,23,24,18,7,12,16,7,26,21,16,21
    .db 7,14,11,15,22,21,26,10,25,25,21,19,23,11,13,9
    .db 28,27,6,17,21,25,6,23,8,18,25,29,9,12,20,12
    .db 21,26,22,14,17,19,20,13,14,12,27,7,18,10,12,17
    .db 7,13,29,23,8,6,17,12,19,13,11,24,20,27,6,12
    .db 24,24,22,11,13,12,20,23,19,17,26,8,23,8,20,25
```