```
;-------------------------------------------------------------------------------
; Titel         :   DOGM162W-A (CGRAM)
;               :
; Funktion      ;   Frage:  Wie kann das CGRAM des LCD DOGM162W-A beschrieben und wie
;               ;           können die Zeichen des CGRAM angezeigt werden?
;               ;
; Bemerkungen   ;   1. Nur 6 der 8 Zeichen des CGRAM werden in diesem Programm benutzt
;               ;   2. Dieses Programm funktioniert mit einem LCD CP594V-0 (mit den
;               ;      erforderlichen Anpassungen, z.B. in LCD_init).
;               ;   3. Controller des LCD DOGM162W-A: ST7036
;               ;      Controller des LCD CP594V-0:   HD44780
;               ;
; Schaltplan    ;   1. Entire Diagram.pdf
;               ;   Für dieses Programm ist nur die Verdrahtung
;               ;   zwischen dem LCD Display und der MPU sowie die +5V Versorgungs-
;               ;   spannung relevant.
;               :
; gewünschte    ;
; Anzeige       ;   5. LCD_line2.pdf
;               ;
; Beispiel mit  ;
; LCD CP594V-0  :   DSC22492.pdf Versuchsaufbau auf einem Steckbrett
;-------------------------------------------------------------------------------
; Processor     :   ATmega8A-PU
; Takt (Quarz)  :   4.096 MHz
; Language      :   Assembler Studio 4
; Date          :   12.08.2014
; Version       :   1.0
; Autor         :   KlausD
;-------------------------------------------------------------------------------
        .include    "m8def.inc"
;-------------------------------------------------------------------------------
; Reset and Interrupt Vector                Description

Begin:  rjmp    Main                        ;  1 POWER ON RESET
        reti                                ;  2 Int0-Interrupt
        reti                                ;  3 Int1-Interrupt
        reti                                ;  4 TC2 Compare Match
        reti                                ;  5 TC2 Overflow
        reti                                ;  6 TC1 Capture
        reti                                ;  7 TC1 Compare Match A
        reti                                ;  8 TC1 Compare Match B
        reti                                ;  9 TC1 Overflow
        reti                                ; 10 TC0 Overflow
        reti                                ; 11 SPI, STC Serial Transfer Complete
        reti                                ; 12 UART Rx complete
        reti                                ; 13 UART Data Register Empty
        reti                                ; 14 UART Tx complete
        reti                                ; 15 ADC Conversion Complete
        reti                                ; 16 EEPROM Ready
        reti                                ; 17 Analog Comparator
        reti                                ; 18 TWI (I²C) Serial Interface
        reti                                ; 19 Store Program Memory Redy

;-------------------------------------------------------------------------------
; Start, Power ON, Reset, Ports

Main:
        ldi     R16     ,   LOW (RAMEND)    ; for  Stackpointer LOW
        out     SPL     ,   R16             ; INIT Stackpointer LOW

        ldi     R16     ,   HIGH(RAMEND)    ; for  Stackpointer HIGH
        out     SPH     ,   R16             ; INIT Stackpointer HIGH

        ldi     R16     ,   0b00111111      ; PORTB Bits 0 until 5 = Output
        out     DDRB    ,   R16             ; PORTB Bits 6 and   7 = Quartz
        ldi     R16     ,   0b00000000      ; PORTB Bits 0 until 7 = 0
        out     PORTB   ,   R16             ;

        ldi     R16     ,   0b00000000      ; PORTC Bits 0 until 7 = Input
        out     DDRC    ,   R16             ;
        ldi     R16     ,   0b00001111      ; PORTC Bits 0 until 3 = 1 (Pullup)
        out     PORTC   ,   R16             ; PORTC Bit  4 = ADC Input for DC
                                            ; PORTC Bit  5 = ADC Input for AC
                                            ; Pullup would adulerate the Measuring Value !

        ldi     R16     ,   0b11111111      ; PORTD Bits 0 until 7 = Output
        out     DDRD    ,   R16             ;
        ldi     R16     ,   0b00000000      ; PORTD Bits 0 until 7 = 0
        out     PORTD   ,   R16             ;

        rcall   LCD_init
        rcall   LCD_clear
        rcall   CGRam

;-------------------------------------------------------------------------------
; Main Loop
```

```
Loop:
        rcall   LCD_line1
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   'L'
        rcall   LCD_data
        ldi     R16     ,   'C'
        rcall   LCD_data
        ldi     R16     ,   'D'
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   'L'
        rcall   LCD_data
        ldi     R16     ,   'i'
        rcall   LCD_data
        ldi     R16     ,   'n'
        rcall   LCD_data
        ldi     R16     ,   'e'
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   '1'
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data

;---------------------------------------------------------------------------------------
; LCD_line2 vorbereiten

        ldi     R16     ,   64              ; = LCD_line2, 1. Zeichen von links
        rcall   LCD_goto

        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   'C'
        rcall   LCD_data
        ldi     R16     ,   'H'
        rcall   LCD_data
        ldi     R16     ,   'R'
        rcall   LCD_data
        ldi     R16     ,   '('
        rcall   LCD_data
        ldi     R16     ,   '0'
        rcall   LCD_data

        ldi     R16     ,   72
        rcall   LCD_goto

        ldi     R16     ,   ')'
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   '='
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data
        ldi     R16     ,   '''
        rcall   LCD_data

        ldi     R16     ,   78
        rcall   LCD_goto
        ldi     R16     ,   '''
        rcall   LCD_data
        ldi     R16     ,   ' '
        rcall   LCD_data

Loop1:
;---------------------------------------------------------------------------------------
; CGRAM '0' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
```

```
        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   0               ; CHR(000) = CGRAM '0'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CGRAM '1' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
        ldi     R16     ,   49              ; CHR(049) = '1'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   1               ; CHR(001) = CGRAM '1'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CGRAM '2' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
        ldi     R16     ,   50              ; CHR(050) = '2'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   2               ; CHR(002) = CGRAM '2'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CGRAM '3' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
        ldi     R16     ,   51              ; CHR(051) = '3'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   3               ; CHR(003) = CGRAM '3'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CGRAM '4' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
        ldi     R16     ,   52              ; CHR(052) = '4'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   4               ; CHR(004) = CGRAM '4'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CGRAM '5' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
        ldi     R16     ,   53              ; CHR(053) = '5'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
```

```
        ldi     R16     ,   5               ; CHR(004) = CGRAM '5'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CHR(048) = '0' anzeigen (diese und folgende dienen zur Funktionskontrolle)

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   52              ; CHR(052) = '4'
        rcall   LCD_data
        ldi     R16     ,   56              ; CHR(056) = '8'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CHR(049) = '1' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   52              ; CHR(052) = '4'
        rcall   LCD_data
        ldi     R16     ,   57              ; CHR(057) = '9'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   49              ; CHR(049) = '1'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CHR(050) = '2' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   53              ; CHR(053) = '5'
        rcall   LCD_data
        ldi     R16     ,   48              ; CHR(048) = '0'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   50              ; CHR(050) = '2'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; CHR(051) = '3' anzeigen

        ldi     R16     ,   70
        rcall   LCD_goto
        ldi     R16     ,   53              ; CHR(053) = '5'
        rcall   LCD_data
        ldi     R16     ,   49              ; CHR(049) = '1'
        rcall   LCD_data

        ldi     R16     ,   77
        rcall   LCD_goto
        ldi     R16     ,   51              ; CHR(049) = '2'
        rcall   LCD_data
        ldi     R16     ,   200             ; ca. 2 Sekunden
        rcall   Wait

;-------------------------------------------------------------------------------
; Jetzt immer wieder dasselbe

        rjmp    Loop1

;-------------------------------------------------------------------------------
; Warte-Routinen
;-------------------------------------------------------------------------------
; Wartezeit 0.1 ms bei 4 MHz
wait01ms:
        push    R16
        push    R17
        ldi     R17     ,   1               ; für 0.1 ms = 1
w01ms1:
```

```
        ldi     R16     ,   123
w01ms2:
        dec     R16
        brne    w01ms2
        dec     R17
        brne    w01ms1
        pop     R17
        pop     R16
        ret

;-----------------------------------------------------------------------------------
; Wartezeit 5 ms bei 4 MHz
wait5ms:
        push    R16
        push    R17
        ldi     R17     ,   50              ; für 1 ms = 10
w5ms1:
        ldi     R16     ,   123
w5ms2:
        dec     R16
        brne    w5ms2
        dec     R17
        brne    w5ms1
        pop     R17
        pop     R16
        ret

;-----------------------------------------------------------------------------------
; Wartezeit: R16 = 1 ==> 0.01 s, R16 = 255 ==> 2.55 s bei = 3.686 MHz

Wait:
        push    R17
        push    R18
        cpi     R16 ,   0
        breq    WLoop0
WLoop1:
        ldi     R17     ,   0b01101110
WLoop2:
        ldi     R18     ,   0b01101110
WLoop3:
        dec     R18
        brne    WLoop3
        nop
        nop
        dec     R17
        brne    WLoop2
        dec     R16
        brne    WLoop1
WLoop0:
        pop     R18
        pop     R17
        ret

;-----------------------------------------------------------------------------------
; LCD-Routinen
;-----------------------------------------------------------------------------------
;****************************************************************** Anfang Initialisierung
; Initialisierung des LCD DOGM162W-A mit ST7036 Dot Matrix LCD Controller/Driver

LCD_init:
        push    R18
        ldi     R18     ,   50
PowerUpWait:
        rcall   wait5ms                     ; Power Up Wait Time
        dec     R18                         ; 250 ms
        brne    PowerUpWait                 ;
        pop     R18
;-----------------------------------------------------------------------------------
        ldi     R16     ,   0b00110000      ; 8 Bit Mode
        swap    R16                         ; 0b00000011 wegen 4 Bit Verdrahtung
        out     PORTD   ,   R16             ; must be set 3 times, why?
;-----------------------------------------------------------------------------------
; Function Set 01: Das 1. Mal
        rcall   LCD_enable                  ; Impuls von E = MPU D.6 Pin 12
        rcall   wait5ms                     ; Wait 5ms
;-----------------------------------------------------------------------------------
; Function Set 02: Das 2. Mal
        rcall   LCD_enable                  ; Impuls von E = MPU D.6 Pin 12
        rcall   wait5ms                     ;
;-----------------------------------------------------------------------------------
; Function Set 03: Das 3. Mal
        rcall   LCD_enable                  ; Impuls von E = MPU D.6 Pin 12
        rcall   wait5ms                     ;
;-----------------------------------------------------------------------------------
; Function Set 04: Den 4-Bit Mode einschalten
        ldi     R16     ,   0b00100000      ; 4 Bit Mode, D.4 = Low = Command
        swap    R16                         ; 0b00000010 wegen 4 Bit Verdrahtung
```

```
        out     PORTD   ,   R16             ;
        rcall   LCD_enable                  ; Impuls von E = MPU D.6 Pin 12
        rcall   wait5ms                     ;
;-----------------------------------------------------------------------------------
; Function Set 05: 4-Bit Mode, 2 Lines, 5x8 Dots, Extension Instructions
        ldi     R16     ,   0b00101001
        rcall   LCD_cmd
;-----------------------------------------------------------------------------------
; Function Set 06: Bias, 2 Lines
        ldi     R16     ,   0b00010100      ; BIAS
        rcall   LCD_cmd                     ;
;-----------------------------------------------------------------------------------
; Function Set 07: Contrast Set (Low Byte)
        ldi     R16     ,   0b01111000      ; Contrast Set
        rcall   LCD_cmd
;-----------------------------------------------------------------------------------
; Function Set 08: ICON/Power/Contrast (High Byte)
        ldi     R16     ,   0b01011110      ; ICON/Power/Contrast
        rcall   LCD_cmd                     ;
;-----------------------------------------------------------------------------------
; Function Set 09: Follower Control
        ldi     R16     ,   0b01101010      ; Follower Control
        rcall   LCD_cmd                     ;
;-----------------------------------------------------------------------------------
; Function Set 10: Display On
        ldi     R16     ,   0b00001100      ; Display On
        rcall   LCD_cmd                     ;
;-----------------------------------------------------------------------------------
; Function Set 11: Clear Display
        ldi     R16     ,   0b00001101      ; Clear Display
        rcall   LCD_cmd                     ;
;-----------------------------------------------------------------------------------
; Functin Set 12: Entry Mode Set
        ldi     R16     ,   0b00001100      ; Entry Mode Set
        rcall   LCD_cmd
; The End
        ret

;******************************************************************* Ende Initialisierung
;-----------------------------------------------------------------------------------
; Ausgabe von Daten zur Anzeige im LCD, sendet ein Datenbyte

LCD_data:
        push    R17
        mov     R17     ,   R16             ; "Sicherungskopie" für das 2. Nibble
        swap    R16                         ; Nibbles vertauschen
        andi    R16     ,   0b00001111      ; obereses Nibble auf Null setzen
        sbr     R16     ,   0b00010000      ; RS auf 1
        out     PORTD   ,   R16             ; 1. Nibble plus RS ausgeben
        rcall   LCD_enable                  ; 1. Nibble plus RS übernehmen
                                            ; 2. Nibble kein swap, da es in R17 schon an
                                            ; der richtigen Stelle steht
        andi    R17     ,   0b00001111      ; oberes Nibble auf Null setzen
        sbr     R17     ,   0b00010000      ; RS auf 1
        out     PORTD   ,   R17             ; 2. Nibble plus RS ausgeben
        rcall   LCD_enable                  ; 2. Nibble plus RS übernehmen
        rcall   LCD_busy                    ; Busy Flag prüfen
        pop     R17
        ret

;-----------------------------------------------------------------------------------
; Ausgabe von Kommandos ans LCD, wie LCD_data aber RS = 0

LCD_cmd:
        push    R17
        mov     R17     ,   R16             ; "Sicherungskopie" für das 2. Nibble
        swap    R16                         ; Nibbles vertauschen
        andi    R16     ,   0b00001111      ; obereses Nibble auf Null setzen
        sbr     R16     ,   0b00000000      ; RS auf 0 (bzw. nicht auf 1)
        out     PORTD   ,   R16             ; 1. Nibble ausgeben
        rcall   LCD_enable                  ; 1. Nibble übernehmen
                                            ; 2. Nibble kein swap, da es in R17 schon an
                                            ; der richtigen Stelle steht
        andi    R17     ,   0b00001111      ; oberes Nibble auf Null setzen
        sbr     R17     ,   0b00000000      ; RS auf 0 (bzw. nicht auf 1)
        out     PORTD   ,   R17             ; 2. Nibble ausgeben
        rcall   LCD_enable                  ; 2. Nibble übernehmen
        rcall   LCD_busy                    ; Busy Flag prüfen
        pop     R17
        ret

;-----------------------------------------------------------------------------------
LCD_enable:
        sbi     PORTD   ,   6               ; Enable High
        nop                                 ; kurz warten
        nop
        nop
```

```
        cbi     PORTD   ,   6               ; Enable Low
        ret

;-----------------------------------------------------------------------------
; Busy Flag prüfen

LCD_busy:
        push    R16
        ldi     R16     ,   0b11110000      ; Disable Data Bit Outputs
        out     DDRD    ,   R16
        ldi     R16     ,   0b00000000      ; Clear all outputs
        out     PORTD   ,   R16
LCD_busy1:
        ldi     R16     ,   0b00100000      ; Enable only read bit
        out     PORTD   ,   R16
        sbi     PORTD   ,   6               ; Raise the Enable signal
        nop                                 ; kurz warten
        nop
        in      R16     ,   PIND            ; Read the current values
        cbi     PORTD   ,   6               ; Disable the Enable signal
        rcall   LCD_enable                  ; Puls the Enable (the second nibble is discarded)
        swap    R16                         ; Busy flag von R16.3 nach R16.7
        sbrc    R16     ,   7               ; Check busy flag
        rjmp    LCD_busy1                   ;
        ldi     R16     ,   0b11111111      ; Enable all outputs
        out     DDRD    ,   R16
        pop     R16
        ret

;-----------------------------------------------------------------------------
LCD_clear:
        ldi     R16     ,   0b00000001      ; Display löschen
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
LCD_home:
        ldi     R16     ,   0b00000010      ; Display Cursor HOME
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
LCD_off:
        ldi     R16     ,   0b00001000
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
LCD_on:
        ldi     R16     ,   0b00001110
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
LCD_line1:
        ldi     R16     ,   0b10000000      ; DRAM auf Adresse 0x00
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
LCD_line2:
        ldi     R16     ,   0b11000000      ; DRAM auf Adresse 0x40
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
; Goto R16 = Adresse (Zeile 1 = 0x00..0x0F, Zeile 2 = 0x40..0x4F
;                                  0....15             64....79
LCD_goto:
        ori     R16     ,   0b10000000      ; Goto DRAM auf Adresse R16
                                            ; ORI - Logical OR with Immediate
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
LCD_CUL:
        ldi     R16     ,   0b00010000      ; Cursor um eine Poation nach links
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
LCD_CUR:
        ldi     R16     ,   0b00010100      ; Cursor um eine Postion nach rechts
        rcall   LCD_cmd
        ret

;-----------------------------------------------------------------------------
```

```
; Zeichen ins CGRAM des LCD schreiben
;-------------------------------------------------------------------------------------

CGRam:
        ldi     ZL      ,   LOW(Symbole*2)
        ldi     ZH      ,   HIGH(Symbole*2)
CGRam1:
        lpm     R16     ,   Z+
        cpi     R16     ,   253
        breq    CGRam2
        rcall   LCD_cmd
        lpm     R16     ,   Z+
        cpi     R16     ,   253
        breq    CGRam2
        rcall   LCD_data
        rjmp    CGRam1
CGRam2:
        ret

;-------------------------------------------------------------------------------------
; Zeichen fürs CGRAM des LCD

Symbole:
        .db 64,0,65,0,66,0,67,0,68,0,69,0,70,0,71,0
        .db 72,0,73,0,74,0,75,16,76,16,77,16,78,0,79,0
        .db 80,0,81,0,82,0,83,24,84,24,85,24,86,0,87,0
        .db 88,0,89,0,90,0,91,28,92,28,93,28,94,0,95,0
        .db 96,0,97,0,98,0,99,30,100,30,101,30,102,0,103,0
        .db 104,0,105,0,106,0,107,31,108,31,109,31,110,0,111,0,253,0

;-------------------------------------------------------------------------------------
```