Friedrich-Ebert-Schule Esslingen	MIKROCONTROLLER	Name:
2.3.2	Wichtige Elemente in Assembler-Dateien	Datum:

1. **Kommentare** in die Quelldatei einfügen!
   Kommentare in Assembler-Dateien werden mit Semikolon (;) eingeleitet. Der Editor stellt den nachfolgenden Text (bis zum Ende der Zeile) **grün** dar.

c:\controller\ed2\datenlogger\adwandler.a51*

```
;*****************************************************************
;* Modul: adwandler.a51                                          *
;*  AD-Wandler Service-Routinen                                  *
;*****************************************************************
;*   Unterprogramme                                              *
;*---------------------------------------------------------------*
;*   ADIN:   Liest einen Wandlerwert (11 Bit) vom ADU (LTC1292)  *
;*           und schreibt ihn in die Speicherstellen 74h und 73h *
;*           Verwendete Register:   r2; r3                       *
;*           Verwendeter Speicher:  DHigh = 74h; DLow = 73h      *
;*---------------------------------------------------------------*
;*   MEAN:   Berechnet den Mittelwert aus zwei Wandlerwerten     *
;*           und schreibt ihn in die Speicherstellen 74h und 73h *
;*           Verwendete Register:   r2; r3                       *
;*           Verwendeter Speicher:  DHigh = 74h; DLow = 73h      *
;*****************************************************************
;* Rolf Rahm, 3.1.06                                             *
;*****************************************************************
   include  c51rd2.inc

   code                  ; Module werden vom Linker an die nächste
                         ; freie Stelle im Code Segment gelegt
```

Der **Programmkopf** sollte Informationen über alle im Programm enthaltenen Unterprogramme enthalten.
⚠ Wichtig sind auch Angaben zum Autor und der letzten Änderung!

Kommentare können an jeder beliebigen Stelle im Quellcode stehen. Eine gute Kommentierung erhöht die Lesbarkeit und die Wartbarkeit des Codes, insbesondere bei grösseren Projekten.
⚠ Wichtige Codeabschnitte sollten immer kommentiert werden!

2. **Kontrollanweisungen**, **Adresssymbole** und **numerische Symbole**

c:\...\adwandler....

```
include   c51rd2.inc

code at 0

CS      bit    P1.4
CLK     bit    P1.7
DOUT    bit    P1.5

DHigh   data   74h
DLow    data   73h

Offset  equ    30d


Bitmaske  equ  01101110b
```

Kontrollanweisung: Einbinden der Symboldefinitionsdatei. Alle Register- und SFR-Bitadressen erhalten symbolische Namen!

Programm steht im Programmspeicher (**Code**-Segment) ab der Adresse 0000h

Bit 4 von Port 1 erhält die symbolische Adresse **CS**
Diese kann in allen Bitbefehlen verwendet werden.
Bsp.: setb CS ; setzen von P1.4

Die Adresse 74h im internen RAM (Segment **data**) erhält das Adresssymbol **DHigh**
Bsp.: mov DHigh,#10 ; [DHigh] ← 10

Das Symbol **Offset** erhält den numerischen Wert 30 (dezimal)
Bsp.: mov DHigh,#Offset ; [DHigh] ← 30

Das Symbol **Bitmaske** erhält den Wert 01101110 (binär)
Bsp.: ANL P2,#Bitmaske ; Alle Bits von Port 2 die in der
; Bitmaske 0 sind werden
; gelöscht. Alle Bits mit 1 in
; der Bitmaske behalten ihren
; Zustand!

3. **Labels**

c:\...\ad...

```
ADIN:  push   0h
       mov    r0,#1
       mov    r2,#0

ad_go:
       setb   DOUT
       clr    CLK
```

Labels sind symbolische Adressen, die als Sprungziele oder für Unterprogrammaufrufe verwendet werden. Sie stehen immer am Zeilenanfang und werden mit einem abschließenden Doppelpunkt (:) gekennzeichnet. Durch **symbolische Adressierung** muss der Programmierer die absolute Sprungadresse beim Programmieren nicht kennen. Diese wird dem Code erst vom Linker zugewiesen!