

**AVR-8-bit-Mikrocontroller**  
**Gruppe 300 - Arbeiten mit AVR-Assembler**  
**Teil 307 - Programmablauf**



### **Teil 301 - Einführung**

- 1 Eine Einführung in die Assembler-Sprache
  - 1.1 Warum Assembler ?
  - 1.2 In der Kürze liegt die Würze - Schnell wie der Wind

### **Teil 302 - Syntax der Assembler-Sprache**

- 2 Struktur und Syntax der Assembler-Sprache
  - 2.1 Struktur der Assembler-Zeilen
- 2.2 Struktur der Programm-Datei
  - 2.3 Kommentare
  - 2.4 Kopf des Programms
  - 2.5 Assembler-Direktive (Directivs)
  - 2.6 Assembler-Ausdrücke (Expressions)
    - 2.6.1 Operanden
    - 2.6.2 Operatoren
    - 2.6.3 Funktionen
    - 2.6.4 Operationen
  - 2.7 Daten- und Adress-Struktur
    - 2.7.1 Nomenklatur: Abkürzungen für Register und Operanden
    - 2.7.2 AVR-Speicher-Räume aus Assembler-Sicht
      - 2.7.2.1 Programm-Speicher
      - 2.7.2.2 Daten-Speicher
      - 2.7.2.3 EEPROM-Speicher
    - 2.7.3 Adress-Modi
      - 2.7.3.1 Direkte Adressierung eines einzelnen Arbeits-Registers
      - 2.7.3.2 Direkte Adressierung von 2 Arbeits-Registern
      - 2.7.3.3 Direkte Adressierung eines I/O-Registers
      - 2.7.3.4 Direkte Daten-Adressierung im SRAM
      - 2.7.3.5 Indirekte Daten-Adressierung mit Displacement
      - 2.7.3.6 Indirekte Daten-Adressierung
      - 2.7.3.7 Indirekte Daten-Adressierung mit Pre-Decrement
      - 2.7.3.8 Indirekte Daten-Adressierung mit Post-Decrement
      - 2.7.3.9 Adressierung des Programm-Speichers
      - 2.7.3.10 Adressierung des Programm-Speichers mit Post-Increment
      - 2.7.3.11 Direkte Adressierung des Programm-Speichers
      - 2.7.3.12 Indirekte Adressierung des Programm-Speichers
      - 2.7.3.13 Relative Adressierung des Programm-Speichers

### **Teil 303 - Instruktionen**

- 3 Mnemotechnische Abkürzungen der Instruktionen
  - 3.1 Arithmetische und logische Instruktionen
  - 3.2 Verzweigungs-Instruktionen
  - 3.3 Instruktionen für den Daten-Transfer
  - 3.4 Bit- und Bit-Test-Instruktionen

### **Teil 304 - Register**

- 4 Register-Datei
  - 4.1 Arbeits-Register
  - 4.2 Unterschiede der Arbeits-Register
  - 4.3 Pointer-Register
  - 4.4 Empfehlungen zu Registerwahl

**AVR-8-bit-Mikrocontroller**  
**Gruppe 300 - Arbeiten mit AVR-Assembler**  
**Teil 307 - Programmablauf**

### **Teil 305 - I/O-Ports**

- 5 Die Ein-/Ausgabe-Ports der AVR's
  - 5.1 Die Ports der AVR's
  - 5.2 Die Ports als I/O-Register
  - 5.3 Das Status-Register SREG

### **Teil 306 - SRAM**

- 6 Statisches RAM - SRAM
  - 6.1 Beschreibung
  - 6.2 Verwendung von SRAM
  - 6.3 Zugriff auf das SRAM
  - 6.4 Verwendung des SRAM als Stack
  - 6.5 Einrichtung des Stapels
  - 6.6 Verwendung des Stapels
  - 6.7 Fehlermöglichkeiten beim (Hoch-)Stapeln

### **Teil 307 - Programmablauf**

- 7 Steuerung des Programmablaufs
  - 7.1 Der Reset
  - 7.2 Linearer Programmablauf und Verzweigungen
  - 7.3 Zeitzusammenhänge beim Programmablauf
  - 7.4 Makros im Programmablauf
  - 7.5 Unterprogramme (Subroutines)
  - 7.6 Interrupts im Programmablauf

### **Teil 308 - Zahlendarstellung**

- 8 Zahlendarstellungen im Assembler
  - 8.1 Dual-Zahlen
  - 8.2 Hexadezimal-Zahlen im ASCII-Format
  - 8.3 Oktal-Zahlen
  - 8.4 BCD-Zahlen (binär kodierte Dezimal-Zahlen)
  - 8.5 Zahlen im ASCII-Format
  - 8.6 Gleitkomma-Zahlen

### **Teil 309 - Bits, Bytes und Zahlen**

- 9 Umwandlungen von Bits, Bytes und Zahlen
  - 9.1 Bitmanipulationen, Schieben und Rotieren
  - 9.2 Umwandlung einer Dual-Zahl in eine ungepackte BCD-Zahl
  - 9.3 Umwandlung einer BCD-Zahl in eine Dual-Zahl
  - 9.4 Umwandlung einer Dual-Zahl in eine Dezimal-Zahl im ASCII-Format
    - 9.4.1 Lösung 1
    - 9.4.2 Lösung 2
    - 9.4.3 Lösung 3
  - 9.5 Umwandlung einer Dezimal-Zahl im ASCII-Format in eine Dual-Zahl
  - 9.6 Umwandlung einer Dual-Zahl in eine Hexadezimal-Zahl im ASCII-Format
  - 9.7 Umwandlung einer Hexadezimal-Zahl im ASCII-Format in eine Dual-Zahl
  - 9.8 Lineare Umwandlung

### **Teil 310 - Anhang und Beispiele**

- 10. Anhang und Beispiele
  - 10.1 Rechner in Assembler
    - 10.1.1 Addition, Subtraktion und Vergleich
    - 10.1.2 Multiplikation
    - 10.1.3 Division

# AVR-8-bit-Mikrocontroller

## Gruppe 300 - Arbeiten mit AVR-Assembler

### Teil 307 - Programmablauf

#### Vorbemerkung

Nichts ist vollkommen - und nichts ist endgültig! So auch nicht dieses Tutorial! Deshalb bitte immer erst nach dem neuesten Datum schauen. Vielleicht gibt es wieder etwas Neues oder eine Fehlerbereinigung oder eine etwas bessere Erklärung. Wer Fehler findet oder Verbesserungen vorzuschlagen hat, bitte melden ([info@alenck.de](mailto:info@alenck.de)).

Immer nach dem Motto: Das Bessere ist Feind des Guten und nichts ist so gut, dass es nicht noch verbessert werden könnte.

Bild-, Beispiel-, Form- und Tabellen-Nummern sind nach folgendem Schema aufgebaut, damit bei Einfügungen/Löschungen nicht alle Nummern wieder geändert werden müssen (hier bunt dargestellt):

Darstellungsart	Abschnitt-LfdNummer: Beschreibung	allgemeines Schema
•	<b>Bild 5.1.4-02: Daten-Adress-Raum</b>	Benummerung eines Bildes
•	<b>Beispiel 5.1.4-03: EEPROM-Speicherung</b>	Benummerung eines Beispiels
•	<b>Form 5.1.3-01: Die main-Funktion</b>	Benummerung einer Formdarstellung
•	<b>Tabelle 5.1.4-01: Schlüsselwörter vom CVAVR</b>	Benummerung einer Tabelle

#### Gravierende Änderungen gegenüber der Vorversion

1.

Völlig neue Strukturierung in **Gruppen** und **Teile**, um das Tutorial umfassend ordnen zu können. Die **Abschnitte** in den **Teilen** sind weitgehend erhalten geblieben.

Gruppenbezeichnung	Kurzbezeichnung
Gruppe 100: Technologie der AVR-8-Bit-Mikrocontroller	Technologie
Gruppe 200: Einsetzen von AVR-Tools	Tools
Gruppe 300: Arbeiten mit AVR-Assembler 3xx_Programm_yy	ASM-Programmierung ASM-Programm-Beispiel
Gruppe 400: ASM-Projekte 4xx_Projekt_zzzz	ASM-Projekte ASM-Projekt-Bezeichnung
Gruppe 500: CodeVisionAVR C-Compiler 5xx_Programm_yy	C-Programmierung C-Programm-Beispiel
Gruppe 600: C-Projekte 6xx_Projekt_zzzz	C-Projekte C-Projekt-Bezeichnung

**xx** steht für die lfd. Nr. des **Teils**, **yy** steht für die lfd. Beispiel-Nr. und **zzzz** steht für die Projekt-Kurz-Bezeichnung

2.

Notwendige Änderungen auf Grund Neuinstallation von **Windows 7**.

3.

**Windows 7** machte eine Installation von **CodeVisionAVR V2.60** als Vollversion notwendig. Daraus leiten sich auch viele Änderungen im Detail für die C-Programmierung (**Gruppe 500**) ab.

4.

Neu-Installation von **AVR Studio Vers. 4.19** unter **Windows 7**

5.

Zur Demonstration des Tools **AVR Studio** ist in **Gruppe 200** eine Trennung in **Teil 205 - Assembler und AVR Studio** und **Teil 206 - C-Compiler und AVR Studio** vorgenommen worden.

6.

**ASM-** und **C-Projekte** werden jeweils in eigenen Gruppen gesammelt (**Gruppe 400** für Assembler- und **Gruppe 600** für C-Projekte).

**AVR-8-bit-Mikrocontroller**  
**Gruppe 300 - Arbeiten mit AVR-Assembler**  
**Teil 307 - Programmablauf**

**IN BEARBEITUNG**