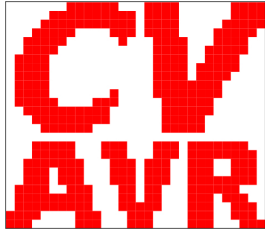


**AVR-8-bit-Mikrocontroller  
Gruppe 500 - CodeVisionAVR C-Compiler  
Inhaltsverzeichnis**



**Gruppe 500 - CodeVisionAVR C-Compiler**

**Teil 501 - Einführung**

- 1 Eine Einführung in **C**
  - 1.1 Warum **C** ?
  - 1.2 Wie entstand **C** ?
  - 1.3 Der AVR-Mikrocontroller in einem eingebetteten System
  - 1.4 Werkzeuge (siehe Gruppe 200)

**Teil 502 - Aufbau eines C-Projektes**

- 2 Was ist ein **C**-Projekt ?
  - 2.1 Erzeugen eines **C**-Projektes
    - 2.1.1 Ein neues Projekt beginnen
    - 2.1.2 Ein **C**-Projekt generieren
  - 2.2 Dateistruktur eines **C**-Projektes
  - 2.3 Einbindung von AVR Studio in den CVAVR
  - 2.4 AVR Studio Debugger
  - 2.5 **C**-Compiler-Optionen

**Teil 503 - Preprozessor**

- 3 Preprozessor-Anweisungen
  - 3.1 Struktur der **C**-Quell-Programme
  - 3.2 **#include**-Anweisung
  - 3.3 **#define**-Anweisung (Makro)
    - 3.3.1 Makros ohne Parameter
    - 3.3.2 Makros mit Parametern
  - 3.4 **#undef**-Anweisung
  - 3.5 **#if**-, **#ifdef**-, **#ifndef**-, **#else**- und **#endif**-Anweisungen
  - 3.6 Andere Preprozessor-Anweisungen

**Teil 504 - Syntax der C-Programmiersprache**

- 4 Die Syntax der **C**-Programmiersprache
  - 4.1 **C**-Quell-Programme
    - 4.1.1 Kommentare
    - 4.1.2 Deklarationen (Vereinbarungen)
    - 4.1.3 Die Funktion **main**
    - 4.1.4 Schlüsselwörter (Keywords) des CodeVisionAVR **C**-Compilers
  - 4.2 Konstanten und Variablen
    - 4.2.1 Zahlensysteme
    - 4.2.2 Datentypen
    - 4.2.3 Konstanten
    - 4.2.4 Variablen
  - 4.3 Operatoren
    - 4.3.1 Arithmetische Operatoren
    - 4.3.2 Relationale Operatoren
    - 4.3.3 Logische und bitweise wirkende Operatoren
    - 4.3.4 Andere Operatoren und Shortcuts
  - 4.4 Komplexe Objekte in **C**
    - 4.4.1 Funktionen
    - 4.4.2 Funktions-Prototypen
    - 4.4.3 Pointers und Arrays
      - 4.4.3.1 Pointers

# AVR-8-bit-Mikrocontroller

## Gruppe 500 - CodeVisionAVR C-Compiler

### Inhaltsverzeichnis

- 4.4.3.1.1 Pointers in Verbindung mit **flash** und **eeprom**
- 4.4.3.1.2 Pointers in Verbindung mit **typedef**
- 4.4.3.2 Arrays
  - 4.4.3.2.1 Ein-dimensionale Arrays
  - 4.4.3.2.2 Zwei-dimensionale Arrays
  - 4.4.3.2.3 Drei-dimensionale Arrays
- 4.4.3.3 Benutzen der Array-Namen als Pointers
- 4.4.3.4 Arrays von Pointers
- 4.4.3.5 Pointers auf Funktionen (Funktionszeiger)
- 4.4.3.6 Funktionen in Verbindung mit **typedef**
- 4.4.4 Strukturen und Unionen
  - 4.4.4.1 Strukturen
  - 4.4.4.2 Unionen
- 4.4.5 Komplexe Typen (eine Zusammenfassung)
- 4.5 Steuerung des Programmablaufs
  - 4.5.1 Anweisungsblöcke { . . . }
  - 4.5.2 Die Anweisung **if**
  - 4.5.3 Die Anweisung **if** in Verbindung mit **else**
  - 4.5.4 Die Fallunterscheidung **switch**
  - 4.5.5 Die Schleife **for**
  - 4.5.6 Die Schleife **while**
  - 4.5.7 Die Schleife **do** in Verbindung mit **while**
- 4.6 Arbeiten mit den Ein-/Ausgabe-Ports

## Teil 505 - Modularer Aufbau der AVR-C-Projekte

### 5 Modularer Aufbau

- 5.1 Das Konzept
- 5.2 Nomenklatur
- 5.3 Die speziellen Header-Dateien
  - 5.3.1 Die spezielle Header-Datei **typedefs.h** (Typ- und Bit-Definitionen)
  - 5.3.2 Die spezielle Header-Datei **iomx.h** (Definitionen aller Register-Bits)
  - 5.3.3 Die spezielle Header-Datei **macros.h** (Definitionen von Makros)
  - 5.3.4 Die spezielle Header-Datei **switches.h** (Definitionen von Schaltern)
- 5.4 Die AVR-C-Module
- 5.5 Anwendung der angepassten AVR-C-Module
  - 5.5.1 Das AVR-C-Modul **application** (Anwendungs-Steuerung)
  - 5.5.2 Das AVR-C-Modul **lcd\_2wire** (Ausgabe auf LCD-20x4)
  - 5.5.3 Das AVR-C-Modul **num\_conversion** (Typ-Konvertierung nach ASCII)
  - 5.5.4 Das AVR-C-Modul **adc\_ref\_1\_1** (ADC mit interner Referenz 1,1 V)
  - 5.5.5 Das AVR-C-Modul **rc5\_decoder** (RC5-IR-Fernsteuerung-Dekoder)
  - 5.5.6 Das AVR-C-Modul **rc5\_encoder** (RC5-IR-Fernsteuerung-Encoder)
  - 5.5.7 Das AVR-C-Modul **usart** (USART-Steuerung)
  - 5.5.8 Das AVR-C-Modul **twi\_master** (I2C- bzw. TWI-Steuerung)
  - 5.5.9 Das AVR-C-Modul **timer0\_pwm** (TIMER0-Steuerung)

## Teil 506 - Anhang

### 6 Anhang

- 6.1 Begriffe und Definitionen
- 6.2 Bibliothek

# AVR-8-bit-Mikrocontroller

## Gruppe 500 - CodeVisionAVR C-Compiler

### Inhaltsverzeichnis

#### 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_yyyyy	ASM-Programmierung ASM-Programm-Beispiel
Gruppe 400: AVR-ASM-Projekte 4xx_Projekt_yyyyy	ASM-Projekte ASM-Projekt-Bezeichnung
Gruppe 500: CodeVisionAVR C-Compiler 5xx_Programm_yyyyy	C-Programmierung C-Programm-Beispiel
Gruppe 600: AVR-C-Projekte 6xx_Projekt_yyyyy	C-Projekte C-Projekt-Bezeichnung

**xx** steht für die laufende Nummer innerhalb des **Teils**, in dem das Programm bzw. das Projekt erscheint und **yyyyy** steht für die Programm- bzw. 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).