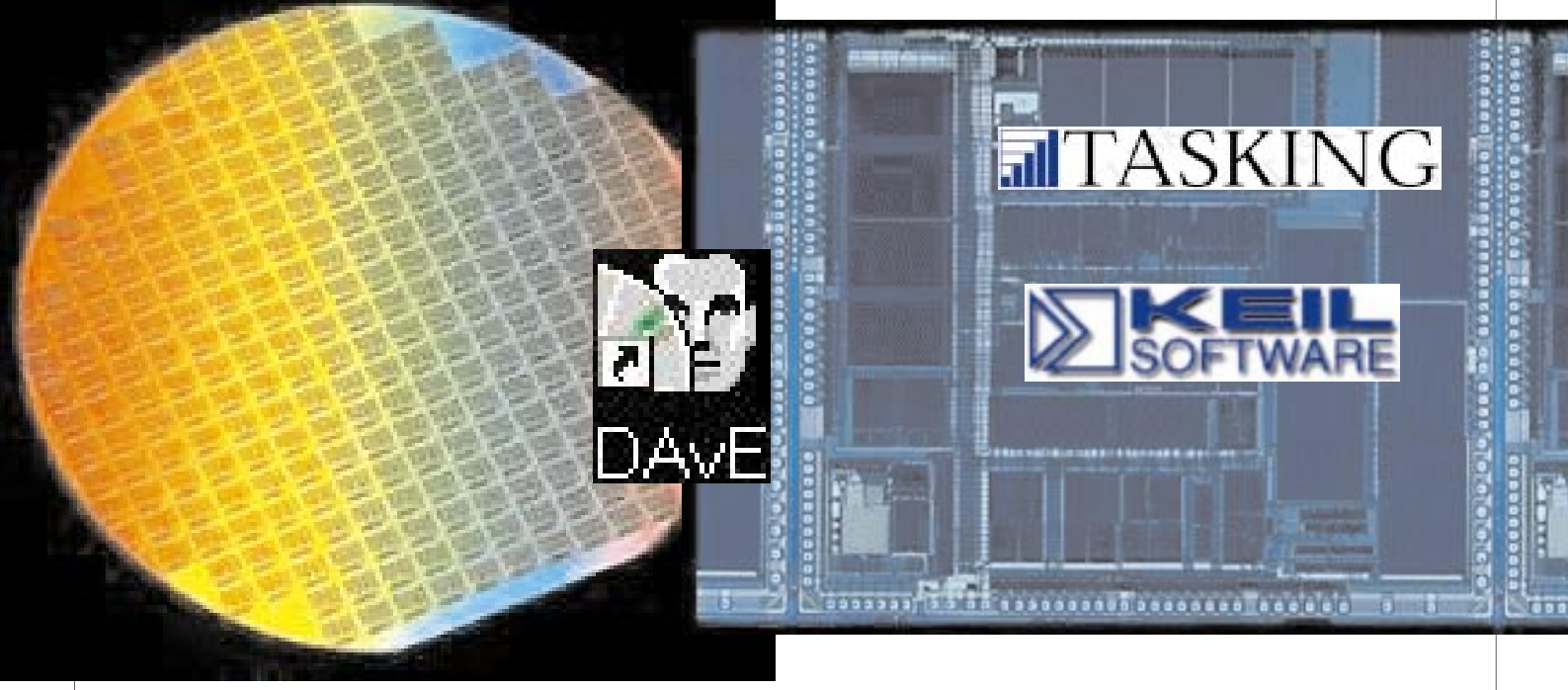


DAvE & Starterkits

Professionelle Softwareentwicklung für Mikrocontroller mit DAvE. Inbetriebnahme des C167CR Starter Kit's.

Dieser Beitrag enthält Hinweise zur Inbetriebnahme des C167CR Starter Kit's. (Generierung der Startup-Datei mit Hilfe von DAvE für die Keil und Tasking Toolkette). Verfasser: Christian Perschl & Wilhelm Brezovits, Tel.: (+43-1) 1707 DW 35883, FAX: (+43-1) 1707 DW 55338, Email: Wilhelm.Brezovits@siemens.at Siemens, B HL/RV, Erdberger Lände 26, A-1030 Wien. Die Verfasser übernehmen keine Gewähr für die Funktionsfähigkeit beschriebener Verfahren, Programme und Schaltungen.

Wilhelm Brezovits, Christian Perschl *http://pcnews.at/ins/57/starterkit/startkit.exe*

1. DAvE - Digitaler Applikationsingenieur

DAvE ist ein Arbeitskollege. Er ist eine CD-ROM. Er ist ein kostenloses Support-Tool.

DAvE bietet eine einzigartige, intuitive Benutzeroberfläche, mit der per Mausklick der ausgewählte 8,16 oder 32 bit Mikrocontroller einfach konfiguriert werden kann.

Durch Drücken der rechten Maustaste erhält man jederzeit Hilfe.

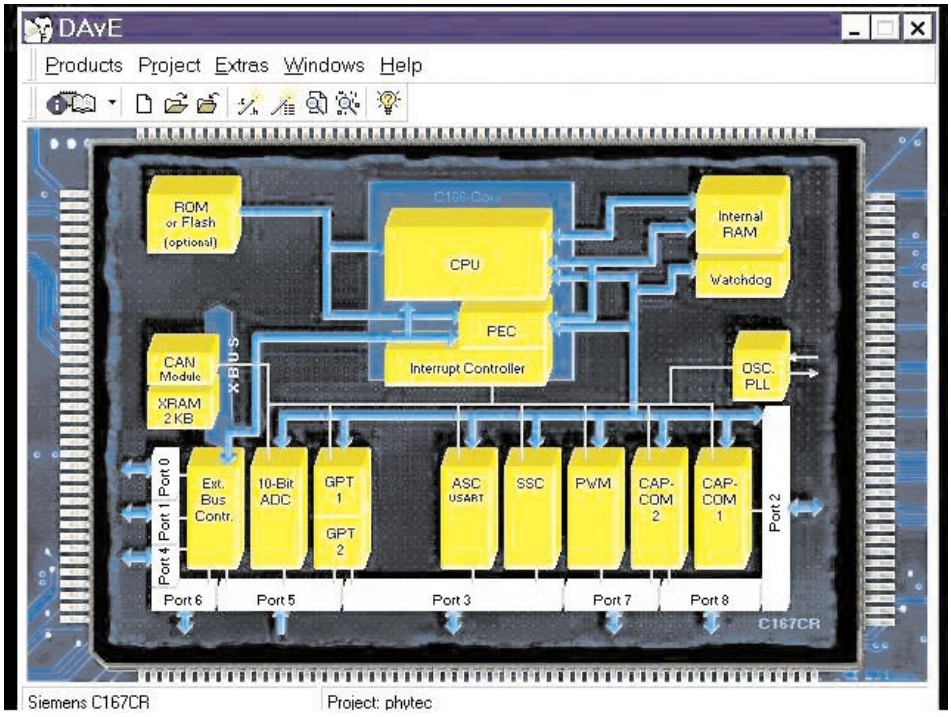
So landet man z. B. bei der Eingabe eines Hex-Wertes durch Drücken der rechten Maustaste im Eingabefeld in einer Binäreingabe.

Weiters erreicht man durch Drücken der rechten Maustaste in einem umrandeten Feld zusätzliche Hilfestellungen. Handelt es sich z. B. um die Konfiguration eines Bits oder eines Bitfeldes, so erhält man eine Darstellung des zugehörigen Registers, oder man springt auf die richtige Seite im Manual.

DAvE generiert automatisch den richtigen C-Code und ein komplettes Dateisystem, welches mit dem File Viewer betrachtet werden kann.

Zwischen "`//USER CODE BEGIN`" und "`//USER CODE END`" besteht die Möglichkeit, eigenen, applikationsspezifischen Source Code einzufügen. Dieser bleibt bei einer Änderung der Konfiguration natürlich erhalten!

DAvE hilft auch bei der Auswahl eines Mikrocontrollers.



NOWATRON

SIEMENS 51

SIEMENS 166

Nach dem Selektieren der Anforderungen liefert DAvE eine Liste aller Mikrocontroller, die die Anforderungen erfüllen (das günstigste Produkt steht dabei an erster Stelle).

2. Starterkits

Starterkits bieten die Möglichkeit, auf günstige Art und Weise einen bestimmten Mikrocontroller und/oder eine Toolkette näher kennenzulernen.

Sie bestehen aus Hard- und Software.

Die Hardware ist ein einschaltfertiges Board (Spannungsstabilisierung, Mikrocontroller - alle Pins zugänglich, Programmspeicher - meist Flash-EEPROM, Datenspeicher - SRAM und Pegeltreiber für RS232). Mit Hilfe mitgelieferter Software kann einfach per Menü im Boot-Mode der Onboard - oder Onchip Programmspeicher programmiert werden. Dazu muß natürlich eine Intel-Hex-Datei vorhanden sein. Diese kann mit der mitgelieferten "Demo"-Software erstellt werden.

Das Wort "Demo" deshalb, da die Compilerpakete eingeschränkt sind (z.B. generierte Codegröβe).

Ist man bereits glücklicher Besitzer einer Toolkette (z.B. Keil oder Tasking-Compiler), so bieten die Starterkits die Möglichkeit, rasch ein neues Derivat der Mikrocontrollerfamilie kennenzulernen ohne zuerst eigene Hardware entwickeln zu müssen.

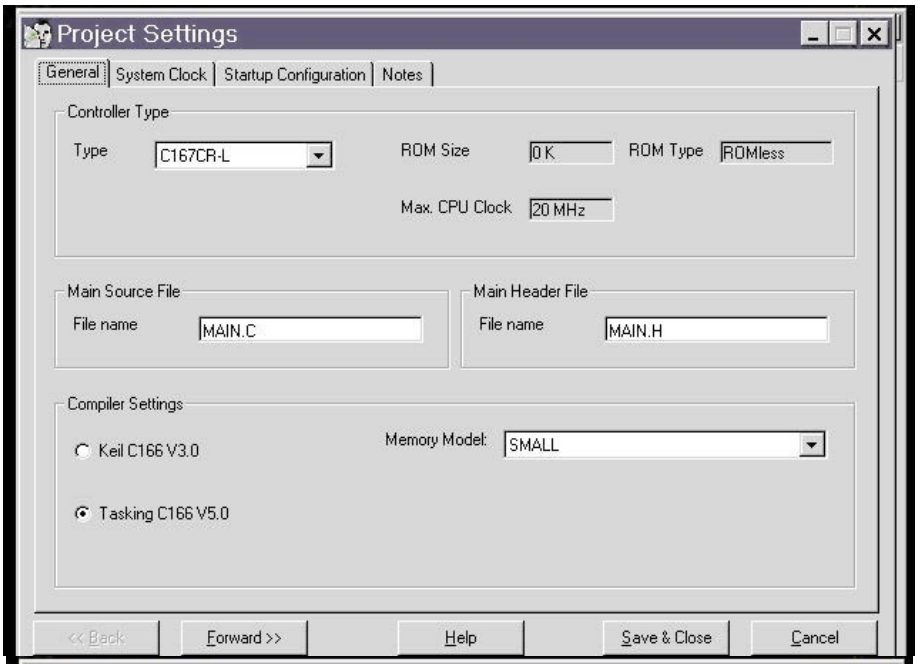
3. Generierung der Startup-Datei mit Hilfe von DAvE für die Keil und Tasking Toolkette

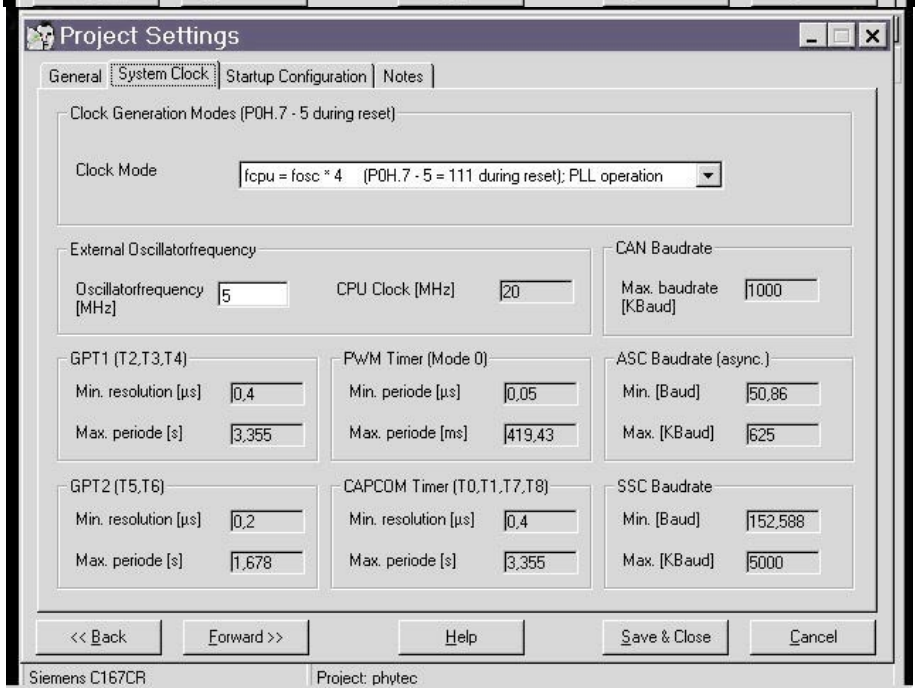
Nach Reset oder "Spannung ein" beginnt der Mikrocontroller mit dem Abarbeiten der internen Resetsequenz. Diese initialisiert alle Special Function Register auf bestimmte Werte. Danach beginnt der Mikroprozessor im Mikrocontroller mit der Abarbeitung des Programms ab Adresse 0.

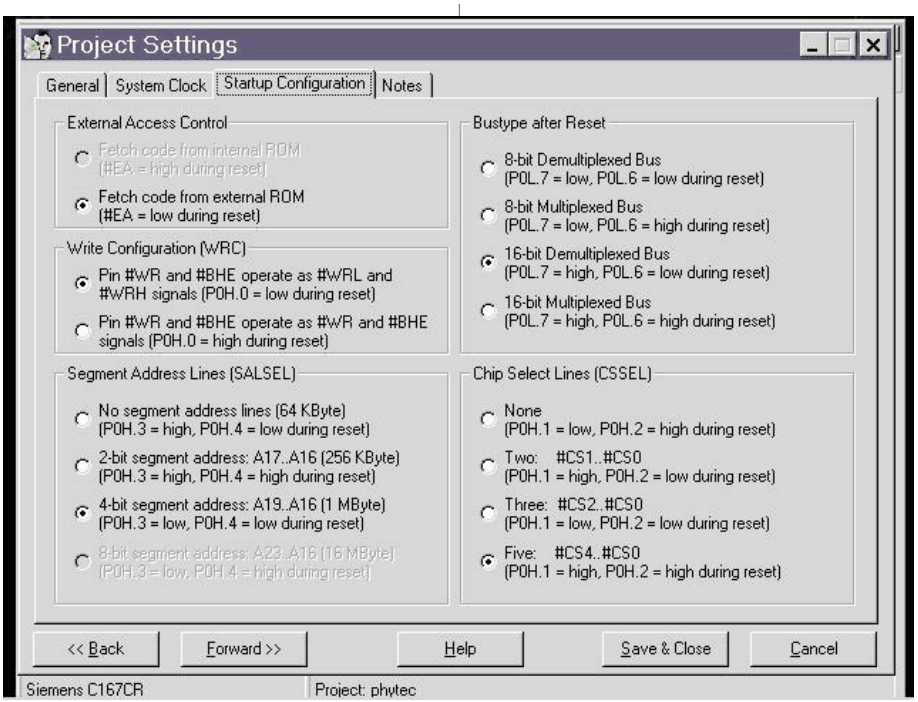
Da bei Adresse 0 die Interrupteinsprungtabelle (Vektortabelle) beginnt befindet sich hier ein Sprungbefehl zum Startup-Code. Der Startup-Code initialisiert den Mikrocontroller (Bus-Timing, Stack-Größe, ...) und schafft die Voraussetzungen für Hochsprachen (Variableninitialisierung, User-Stack, Floating-Point-Stack, ...). Der letzte Befehl des Startup-Code ist ein Sprung zum Hochsprachen-Hauptprogramm `void main (void)`.

Der Startup-Code befindet sich bei der Keil-Toolkette in der Datei `START.ASM` und in der Tasking-Toolkette in der Datei `CSTART.ASM`.

Dave generiert eine Startup-Datei mit dem Namen `START.ASM`.







Zuerst wird mit DAvE ein Projekt (`phytec`) erstellt (`Project, New`).

Bei den ***Project Settings*** stellen wir die verwendete Toolkette (Keil oder Tasking) ein.

Beide Compiler sind zwar ANSI-C Compiler unterscheiden sich aber in der Syntax der mikrocontrollerspezifischen Erweiterungen.

Durch das Einstellen der Taktfrequenz erhalten wir eine Übersicht über die zeitlichen Eigenschaften der On-Chip Peripherals.

In Übereinstimmung mit dem Phytec KitCON-167 Hardware-Manual wird die Startup(Einschalt)-Konfiguration und der External Buscontroller für

BUSCON0 - CS0 - 256 KByte FLASH Bank1 U8/U9 - 00:0000h bis 03:FFFFh und

BUSCON1 - ADDRSEL1 - CS1 - 64 KByte RAM Bank1 U10/U11 - 04:0000h bis 04:FFFFh

konfiguriert.

Bei 55ns Speichern gilt: 0 Waitstate, RW-Delay, no Tri-state, short ALE, 16-Bit-Demultiplexed und Umschaltung von A0/BHE# zu Write LOW (WRL#) und Write HIGH (WRH#) - siehe auch Karl-Heinz Mattheis Buch Seite 94 und Kapitel 4.

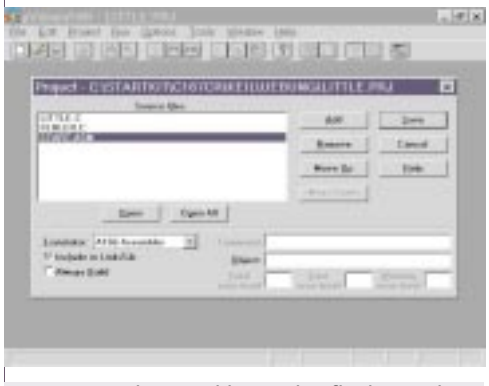
Mit ***Project - Generate Code*** initiieren wir die Source-Code-Generierung die mit ***Project - File Viewer*** betrachtet werden kann.

Hinweis:

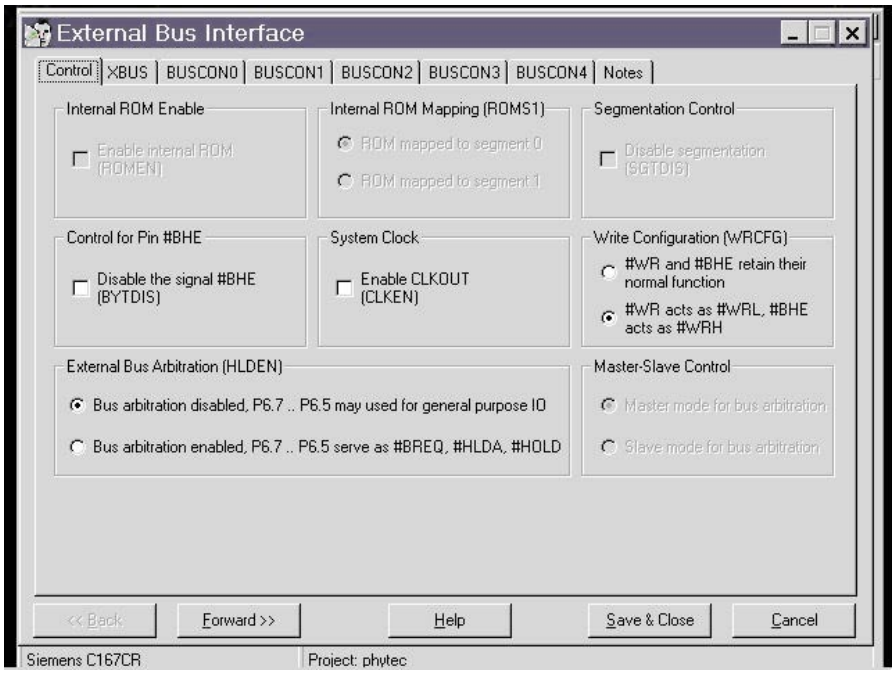
DAvE generiert für die Keil und für die Tasking Toolkette eine Startupdatei mit dem Namen `START.ASM`.

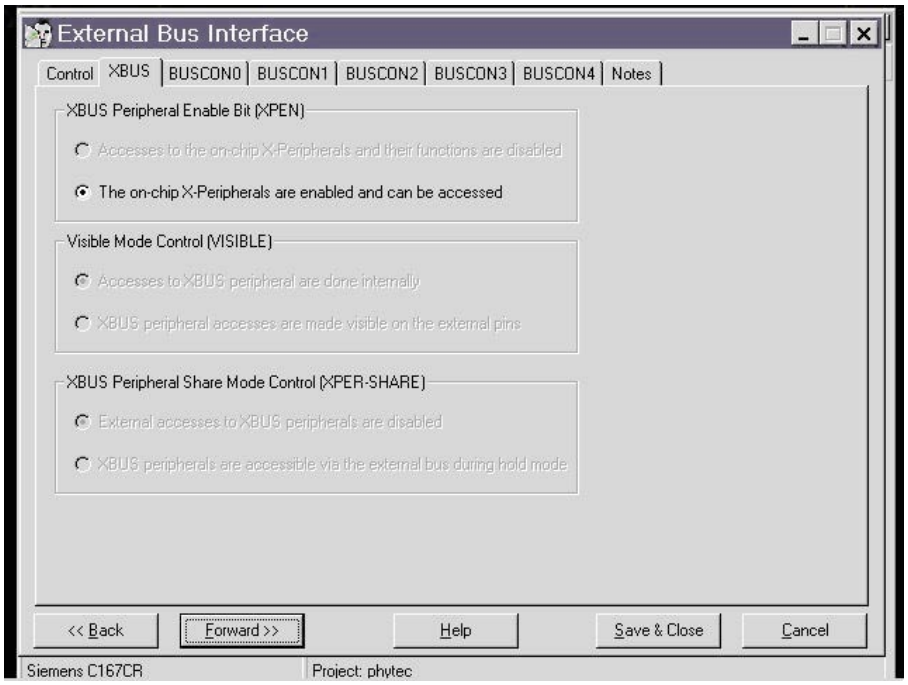
Hinweis für die Keil-Toolkette

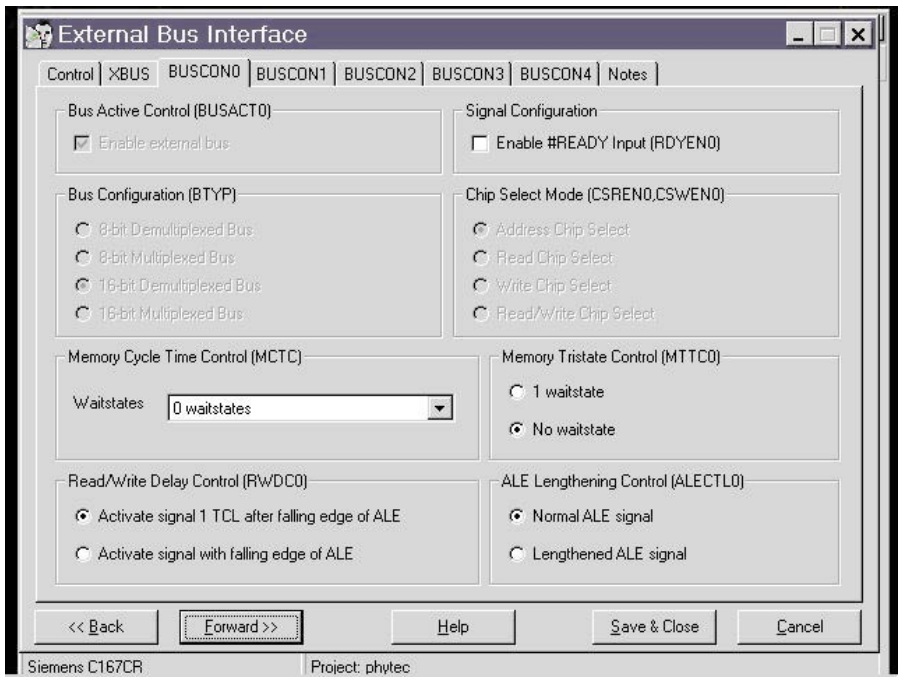
Die Initialisierung des externen Buscontrollers (`BUSCON0` bis `BUSCON4`) wird von DAvE in der Startupdatei gemacht und mit `*** INSERTED ***` gekennzeichnet.



µVision - Keil Entwicklungsoberfläche, Einbindung der Startupdate







Die von DAvE generierte Startupdatei START.ASM für unser C167CR Starterkit wird danach folgendermaßen in die Keil Entwicklungsumgebung µVision eingebunden:

Wichtig dabei ist, daß "***Include in Link/Lib***" angekreuzt ist!

Hinweis für die Tasking-Toolkette:

Wir benennen die Startupdatei start.asm um auf cstart.asm.

Die Initialisierung von BUSCON0 wird von DAvE in der Startupdatei gemacht und mit

*** INSERTED *** gekennzeichnet.

Die Initialisierung von BUSCON1 bis BUSCON4 entnehmen wir der von DAvE generierten Datei main.c (ADDRSEL1 = 0x0404; BUSCON1 = 0x04AF;) und fügen in Assembler-Syntax diese Anweisungen in der Datei cstart.asm ein (siehe inserted by Wilhelm, 29.1.1998):

die Startupdatei sieht dann so aus:

MAKEFILE

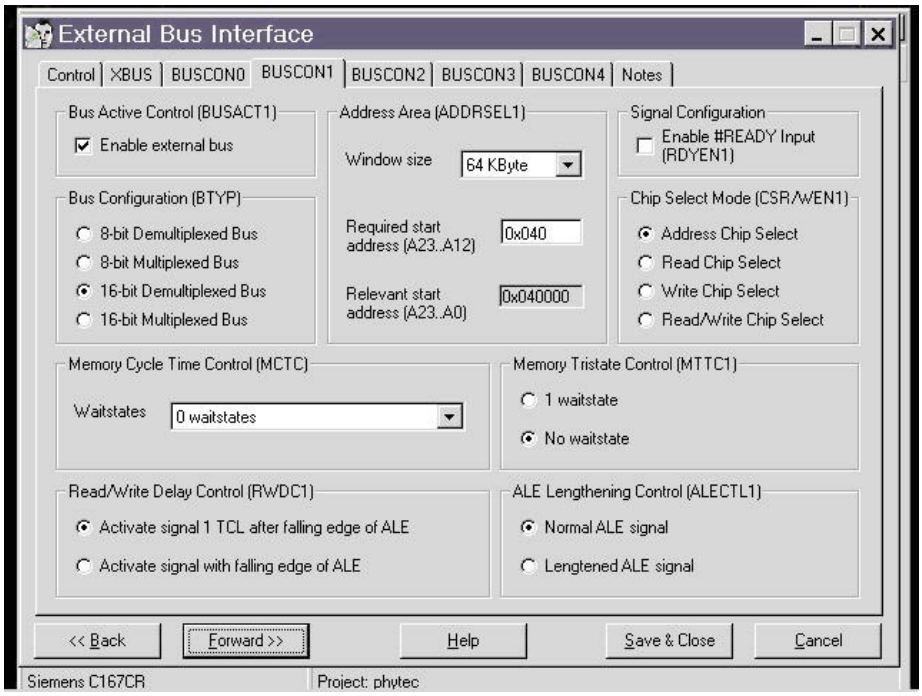
```
# MAKROASSEMBLER
cstart.src: cstart.asm
m166 cstart.asm DEFINE(MODEL, SMALL)
DEFINE(FLOAT) DEFINE(C167) DEFINE(PHYTEC)
```

IV.) Locater Steuerdatei für das C167CR Starterkit:

Die Locater Anweisungen sind notwendig, damit die vom Compiler generierten Segmente (Code, Daten) vom Linker/Locater auf die richtigen physikalischen Speichermedien (ROM, RAM) gemäß der Hardware (C167CR Starterkit) aufgeteilt werden.

a) bei der Keil-Toolkette

µVision - Keil Entwicklungsoberfläche, Linker/Locater




```
;*******************************************************************************
;* __CSTART
;*******************************************************************************
__CSTART_PR SECTION CODE WORD PUBLIC 'CPROGRAM'
__CSTART  PROC TASK __CSTART_TASK INTNO __CSTART_INUM = 00H
 DISWDT    ; Disable watchdog timer.
      ; Set SYSCON register.
 BFLDL SYSCON, #SYSC_M_L, #(SYSC_L AND SYSC_M_L)
 BFLDH SYSCON, #SYSC_M_H, #(SYSC_H AND SYSC_M_H)
 BSET    SYSCON.2                ; Set XPEN *** INSERTED ***
;  BFLDH SYSCON, #SYSC_M_H, #(SYSC_H AND SYSC_M_H)
@IF( @C167 )
                         ;Set BUSCON0 register
 BFLD BUSCON0, #BUSC0_M_L, #(BUSC0_L AND BUSC0_M_L)
 BFLDH  BUSCON0,#BUSC0_M_H,#(BUSC0_H AND BUSC0_M_H)
@ENDI
 MOV STKOV,#?SYSSTACK_BOTTOM+6*2;Set stack underflow pointer.
 MOV STKUN, #?SYSSTACK_TOP  ; Set stack overflow pointer.
 MOV SP, #?SYSSTACK_TOP  ; Set stack pointer.
 MOV CP,  #RBANK  ; Set context pointer.
```

Ausschnitt der Datei CSTART.ASM

b) bei der Tasking-Toolkette

Ausschnitt der Kommandodatei CMD_LOC für den Locater-AufrufMEMORY ROM (0000000h to 000DFFFh) ; 56 KB int. Flash EEPROM

```
MEMORY RAM      ( 000E000h to 000E7FFh) ;       2      KB int. XRAM
RESERVE MEMORY (000E800h to 000EEFFh) ;      1,75   KB Reserved Space
RESERVE MEMORY (000EF00h to 000EFFFh) ;      256     B int. CAN Module
MEMORY RAM      ( 000F000h to 000F1FFh) ;       512     B int. ESFRs
RESERVE MEMORY (000F200h to 000F5FFh) ;      1      KB Reserved Space
MEMORY RAM      (000F600h to 000FDFFh) ;     2      KB int. Dual Port RAM
MEMORY RAM      (000FE00h to 000FFFFh) ;     512     B int. SFR
MEMORY ROM      (0010000h to 003FFFFh) ;     192    KB ext. Flash EEPROM
MEMORY RAM      (0040000h to 004FFFFh) ;     64     KB ext. RAM
;nicht bestückt, nicht erreichbare Speicheradressen:
RESERVE MEMORY (50000H to 0FFFFFFH)
```

5. Hinweis

Rückäußerungen erwünscht.

Die hier besprochenen Dateien für das C167CR Starterkit (Source-Code, Linker-Locater-Steuerung, Projektdatei, makefile, ...) können vom Autor per E-Mail *wilhelm.brezovits@siemens.at* angefordert werden.

SIEMENS-3