6 Banking

6.1 Allgemeines

Übersteigt der von Ihnen benötigte Code den Adreßraum des 8051, muß eine Adreßerweiterung durchgeführt werden. Hierzu werden zusätzliche Codebereiche in den logischen Adreßraum des 8051 eingeblendet. Die Erweiterung des Adreßraums kann mittels Portpins, über ein zusätzliches Latch, das über eine Adresse im xdata-Bereich angesprochen wird oder über ein eigenes Verfahren, das die zusätzlichen Adressen enthält, erfolgen. Das Einblenden von Speicherbereichen wird als Banking bezeichnet. Das Umschalten der Bänke erfolgt mittels Software. Die Bankumschaltung verwaltet mehrere Codebereiche, die in einem definierten Adreßraum des 8051 liegen. Es kann immer nur eine Bank aktiv sein. Die Bankumschaltung muß in einem Common-Adreßbereich liegen. Der Common-Adreßbereich ist von der Bankumschaltung ausgenommen (siehe Abbildung 63).

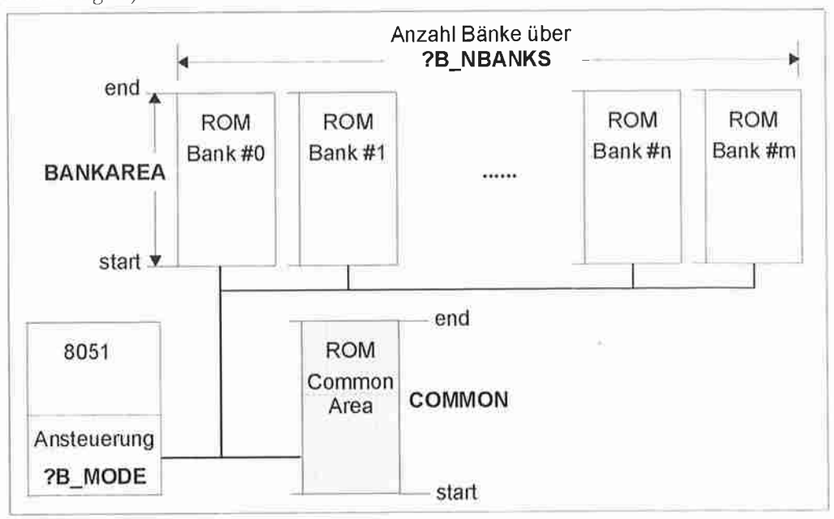


Abbildung 63 Schematischer Aufbau Bankumschaltung

Nach dem Reset ist eine der Bänke aktiv. Wollen Sie eine bestimmte Bank nach dem Reset aktivieren, so sollten Sie dies schon in der Hardware realisieren.

☞ !! In einem gebankten Programm sind nicht alle Funktionen gleichzeitig verfügbar. !!

Umschaltprinzip

Soll nun eine Funktion in einer anderen Bank aufgerufen werden, wird eine Bankumschaltsequenz aufgerufen. In dieser Sequenz steht die Information, in welcher Bank und an welcher Adresse sich die gewünschte Funktion befindet. Eine genaue Beschreibung des Umschaltprinzips des BL51 finden Sie im Kapitel 6.7.

Bevor das eigentliche Umschalten stattfindet, wird die Nummer der aktiven Bank oder die Adresse der Umschaltsequenz (Prinzip BL51 auf dem Stack abgespeichert. Nachdem die Funktion in der Bank abgearbeitet ist, wird die vorhergehende Bank wieder aktiviert und das Programm fortgesetzt. Die Rückschaltung ist deshalb notwendig, weil während des Compilierens nicht bekannt ist, von welcher Bank der Aufruf erfolgte. Die Bankumschaltung muß über eine zusätzliche Adreßleitung durchgeführt werden. Die Adreßerweiterung kann z. B. über Portpins oder über einen Adreßlatch im XRAM-Bereich erfolgen. In Abbildung 64 ist eine Adreßerweiterung über einen Portpin realisiert worden.

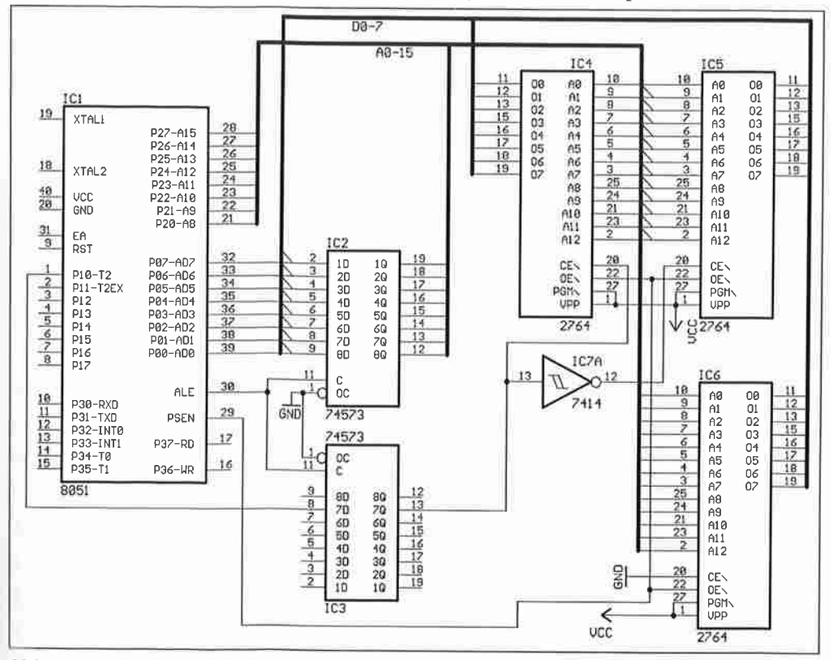


Abbildung 64 Beispiel einer Bankumschaltung mit 2 Bänken

6.2 Banking mit dem BL51

Mit dem BL51 kann das Programm in bis zu 32 Code-Bänke und einem Common-Code-Bereich aufgeteilt werden. Die Common-Code-Bereich ist stets von allen Bänken erreichbar. Der BL51 bietet folgende Umschaltverfahren an:

1. Umschaltung über Portpins des 8051 (z. B. Port 1).
2. Umschaltung über einen Multiplexer, der über eine Adresse im 64kByte-Adreßbereich angesprochen wird.
3. Frei konfigurierbares Umschaltverfahren. Dies könnte z. B. eine Programmiersequenz im EPLD oder eine bestimmte Ansteuerung eines FPGAs sein.

☞ !! Es wird nur eine Bankarea mit 64kByte Adreßraum unterstützt. !!

6.3 Aufbau des Common-Code-Bereichs (Keil)

Dieser Bereich enthält Routinen und Konstanten, die zu jeder Zeit vom Programmablauf erreichbar sein müssen. Dies sind z. B. die Reset-Adresse, die Interrupt-Tabelle, Interrupt-Funktionen, String-Konstanten, Library-Funktionen des C51-Compilers sowie die Routinen für die Bankumschaltung.

☞ !! Ist der für den Common-Code-Bereich vorgesehene Speicherbereich nicht groß genug, dupliziert der BL51 den überhängenden Teil des Common-Code-Bereichs an den Anfang jeder vorhandenen Code-Bank. !!

Der BL51 verfügt über folgende Steueranweisungen für das Banking:

- **BANKAREA**: Angabe des gebankten Adreßbereichs
- **BANKx**: Angabe der verwendeten Bank
- **COMMON**: Angabe des Common-Bereichs

Eine genaue Beschreibung der Steueranweisung finden Sie auf den folgenden Seiten.

```
COMMON(   // Angaben, welche Funktionen im Common-Bereich liegen
?PR?*?MAIN,
?PR?*?ABLAUF,
?CO?TABELLE
)

BANK0(    // Bank 0 enthaelt alle Testsequenzen
?C_C51STARTUP,
?PR?*?TEST
)

Bank1(    // Bank 1 enthaelt alle Steuersequenzen
?PR?*?ABGLEICH,
?PR?*?SCSI,
?PR?*?USB
)

BANKAREA(0x8000,0xBFFF)
```

Listing 76 Beispiel für ein gebanktes Projekt (Auszug LIN-File)

☞ !! Der BL51 ist nicht in der Lage die Bänke selbst zu füllen. !!

BANKAREA (start, end)

- **BANKAREA**: Mit diesem Parameter wird der Bereich angegeben, in der die Bankumschaltung stattfindet. Diese Angaben bilden die physikalischen Adressen der Bankarea. Alle Source-Module, die mit dem Steuerparameter BANKx gekennzeichnet sind, werden in diesem Bereich gelagert.
- start: Startadresse des Bankbereichs
- end: Endadresse des Bankbereichs

☞ !! Wird keine BANKAREA angegeben, verwendet der BL51 standardmäßig die Einstellung: BANKAREA(0H, 7FFFH) !!

☞ !! Belegt die BANKAREA nur einen Zwischenbereich im Adreßraum, ist der nachfolgende Bereich wieder COMMON. !!

Soll die Bankarea unter µVision eingegeben werden, so müssen Sie im Menüpunkt Options\BL51 Code Banking Linker....\Additional folgenden Eintrag vornehmen:

Abbildung 65 Eingabe der Bankarea unter µVision

In welche Bank die einzelnen Codesegmente abgelegt werden sollen, wird über den Parameter BANKx des Linkers angegeben. Bei dem Parameter können Filenamen in {}-Klammern oder Segmentnamen über ()-Klammern angegeben werden.

BANKx {Filename (Segmentname), ...}
BANKx (Segmentstartadresse, Segmentname(Startadresse), ...)

- **BANK**: Mit diesem Parameter wird dem BL51 angegeben, in welcher Bank welches Segment abgelegt werden soll.
- x: x kann den Wert bis n-1?B_NBANKS annehmen. Die erste Bank wird als 0 bezeichnet. Wie viele Bänke bei Ihrer Applikation vorhanden sind, wird über die Konstante ?B_NBANKS im Source-Modul bl51_bank.a51 angegeben.
- {}: Angabe über Filename
- **()**: Angabe über Segmentname

```
BANK0{D:\8051\C51\EXAMPLES\BL51_EX1\C_BANK0.OBJ}, &
BANK1(0x1000h, ?PR?FUNC2?C_BANK2), &
```

Abbildung 66 Angabe einer Bankzuweisung beim BL51

Mit dem Parameter COMMON wird festgelegt, welche Segmente in den COMMON-Bereich abgelegt werden sollen. Zum festen Bestandteil dieses Bereichs gehören:

1. Reseteinsprung
2. Interrupt-Tabelle
3. Bankumschaltung
4. Library-Funktionen des C51-Compilers
5. Konstanten, die im Codebereich liegen

COMMON {Filename (Segmentname), ...}
COMMON (Segmentstartadresse, Segmentname(Startadresse), ...)

- **COMMON**: Die mit COMMON gekennzeichneten Segmente werden in den COMMON-Bereich abgelegt.
- {}: Angabe über Filename
- **()**: Angabe über Segmentname

```
COMMON{D:\8051\C51\EXAMPLES\BL51_EX1\L51_BANK.OBJ}, &
```

Abbildung 67 Angabe der COMMON Zuweisung beim BL51

6.4 Konfiguration der Bankumschaltung

Wenn Sie eine Bankapplikation erstellen wollen, müssen folgende Angaben vorhanden sein:

1. Wie viele Bänke verwendet die Applikation?
2. Welches Verfahren der Umschaltung wird verwendet?

Die Einstellungen für das verwendete Bank-Switch-Verfahren werden im Source-Modul **L51_BANK.a51** vorgenommen. Dieses File enthält Konstanten, über die Sie die Einstellung für das verwendete Verfahren sowie für die Adresse vornehmen können. In Listing 77 ist der Konfigurationsbereich abgebildet.

```
;************* Configuration Section ***************************
?B_NBANKS   SET  16   ; Define max. Number of Banks              *
;                     ; the max. value for ?B_BANKS is 32        *
; Note: Valid numbers for ?B_NBANKS are 2, 4, 8, 16, or 32. If  *
; you have an appl. with 3 banks you must define ?B_NBANKS to 4!*
;                                                                *
?B_MODE     EQU   1   ; 0 for Bank-Switching via 8051 Port      *
;                     ; 1 for Bank-Switching via XDATA Port     *
;                     ; 4 for user-provided bank switch code   *
;                                                               *
?B_RTX      EQU   0   ; 0 for applications without RTX-51 FULL *
;                     ; 1 for applications using RTX-51 FULL    *
;                                                               *
IF  ?B_MODE = 0;                                               *
;---------------------------------------------------------------*
; if ?BANK?MODE is 0 define the following values               *
; For Bank-Switching via 8051 Port define Port Address / Bits  *
?B_PORT     EQU   P1  ; default is P1                          *
?B_FIRSTBIT EQU    3  ; default is Bit 3                       *
;---------------------------------------------------------------*
ENDIF;                                                         *
;                                                              *
IF  ?B_MODE = 1;                                               *
;---------------------------------------------------------------*
; if ?BANK?MODE is 1 define the following values               *
; For Bank-Switching via XDATA Port define XDATA Port Address/ *
;                                                        Bits  *
?B_XDATAPORT EQU 0FFFFH ; default is XDATA Port Address 0FFFFH *
?B_FIRSTBIT  EQU 0      ; default is Bit 0                     *
;---------------------------------------------------------------*
ENDIF;                                                         *
;                                                              *
;***************************************************************
```

Listing 77 Auszug aus l51_bank.a51

?B_MODE

- **?B_MODE**: Mit diesem Bit wird das Verfahren SFR/XDATA Umschaltung selektiert. Ist ?B_MODE = 0, wird eine Umschaltung über einen 8051 Port vorgenommen.
  - ☞ !! Ist B_MODE = 0, gilt die Definition von ?B_Port !!
  - ☞ !! Ist B_MODE = 1, gilt die Definition von ?B_XDATAPORT. !!

?B_NBANKS

- **?B_NBANKS**: Mit dieser Konstanten wird die Anzahl der verwendeten Bänke angegeben. Folgende Werte sind zulässig: 2, 4, 8, 16 und 32.
  - ☞ !! Verwenden Sie z. B. 5 Bänke, **müssen** Sie 8 eingeben. !!

?B_PORT

- **?B_PORT**: Ist B_MODE mit 0 definiert, wird über B_PORT der Port für die Bankumschaltung angegeben. Es sind alle Ports zulässig, die als Ausgabe verwendet werden können.
  - ☞ !! P0 und P2 sind **nicht** zugelassen. !!
  - 💣 !! Die Umschaltung muß so realisiert werden, daß die verwendeten Portpins aufeinander folgen. Wenn dies nicht der Fall ist oder die Portpins über mehrere Ports verstreut liegen, so müssen Sie eine selbstdefinierte Bankumschaltung (siehe Kapitel 6.10) verwenden. !!

?B_XDATAPORT

- **?B_XDATAPORT**: Ist B_MODE mit 1 definiert, wird über diese Konstante die Adresse im XDATA-Bereich angegeben, mit der die Hardwareumschaltung erfolgen soll.

Über die Definition ?B_FIRSTBIT wird die Angabe des Steuerbits vorgenommen.

?B_FIRSTBIT

- **?B_FIRSTBIT**: Mit dieser Definition wird das erste Steuerbit angegeben. Werden mehrere Bits zur Ansteuerung benötigt, so werden diese fortlaufend angelegt.
  - ☞ !! Werden mehr Bits zur Ansteuerung benötigt, als am Port noch vorhanden sind, gibt der Assembler die Warnung 52 aus. !!

Mit der Funktion **switchbank** wird die Bankumschaltung vorgenommen.

```
;***************  SWITCHBANK FUNCTION  ************************
;                                                              *
; SWITCHBANK allows use of bank-switching for C programs       *
;                                                              *
; prototype:   extern switchbank (unsigned char bank_number);  *
;                                                              *
;***************************************************************
_SWITCHBANK: MOV    A,R7
IF ?B_MODE = 0  OR  ?B_NBANKS > 16
             SWAP   A
             RR     A
ENDIF

IF ?B_MODE = 1  AND  ?B_NBANKS <= 16
             SWAP   A
ENDIF
             MOV    DPTR,#?BANK?SWITCH
             JMP    @A+DPTR
```

Listing 78 Umschaltfunktion switchbank()

6.5 Einstellungen unter µVision

Wird ein Projekt angelegt, das eine Bankumschaltung beinhaltet, müssen folgende Einstellungen unter µVision vorgenommen werden:

1. Dem BL51 muß bekanntgegeben werden, daß die Applikation eine Bankumschaltung enthält. Dies erreichen Sie über den Button „Code banking" im Menüpunkt: **Options\BL51 Code Banking Linker..** (siehe Abbildung 68).

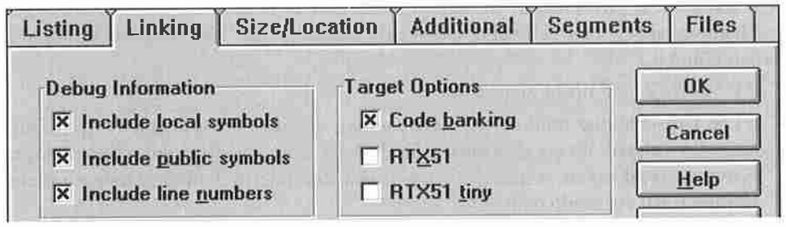


Abbildung 68 Einstellung vom BL51 unter µVision

2. Im Projektfenster wird zu jedem Source-Modul die entsprechende „Bank number" eingetragen (siehe Abbildung 69).

☞ !! Wird keine Nummer angegeben, so wird das Source-Modul in den Common-Bereich abgelegt. !!

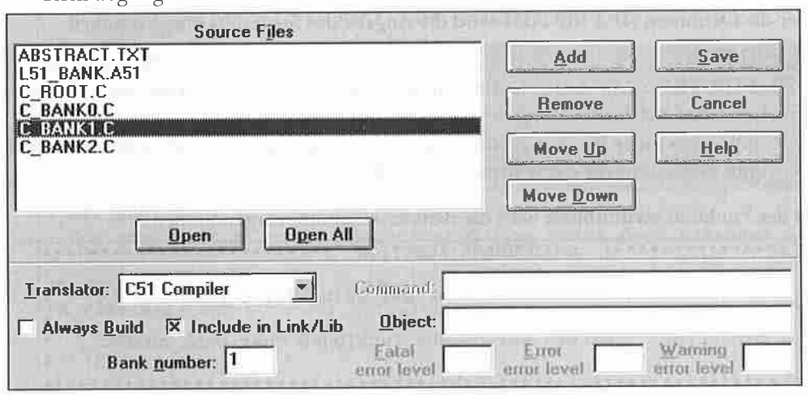


Abbildung 69 Einstellung im Projektfenster

Sie können Ihre Einstellungen nochmals überprüfen, indem Sie beim Menüpunkt **Options\BL51 Code Banking Linker\Files,** den Button **Keep linker response file** anklicken. Mit diesem Menüpunkt erzeugt die Oberfläche ein LNK-File von Ihrem Projekt. Es enthält alle zu compilierenden Source-Module sowie alle in der Oberfläche eingeschalteten Optionen. In Listing 79 ist das LNK-File des Beispiels BL51_EX1.PRJ abgebildet.


```
COMMON{D:\8051\C51\EXAMPLES\BL51_EX1\L51_BANK.OBJ}, &
COMMON{D:\8051\C51\EXAMPLES\BL51_EX1\C_ROOT.OBJ}, &
BANK0{D:\8051\C51\EXAMPLES\BL51_EX1\C_BANK0.OBJ}, &
BANK1{D:\8051\C51\EXAMPLES\BL51_EX1\C_BANK1.OBJ}, &
BANK2{D:\8051\C51\EXAMPLES\BL51_EX1\C_BANK2.OBJ}, &
BANK2{D:\8051\C51\EXAMPLES\BL51_EX1\STARTUP.OBJ} &
TO D:\8051\C51\EXAMPLES\BL51_EX1\BL51_EX1
```

Listing 79 LNK-File des Beispiels BL51_EX1.PRJ

6.6 Einstellungen unter µVision2

Unter µVision2 wird die Bankkonfiguration bei der Erstellung des Projekts durchgeführt. Über den Button „Code Banking" wird das Banking aktiviert. Die Anzahl der Bänke wird über den Menüpunkt „Banks:" ausgewählt. Der Adreßbereich für die Bänke wird in den Feldern **Start** und **End** eingetragen.

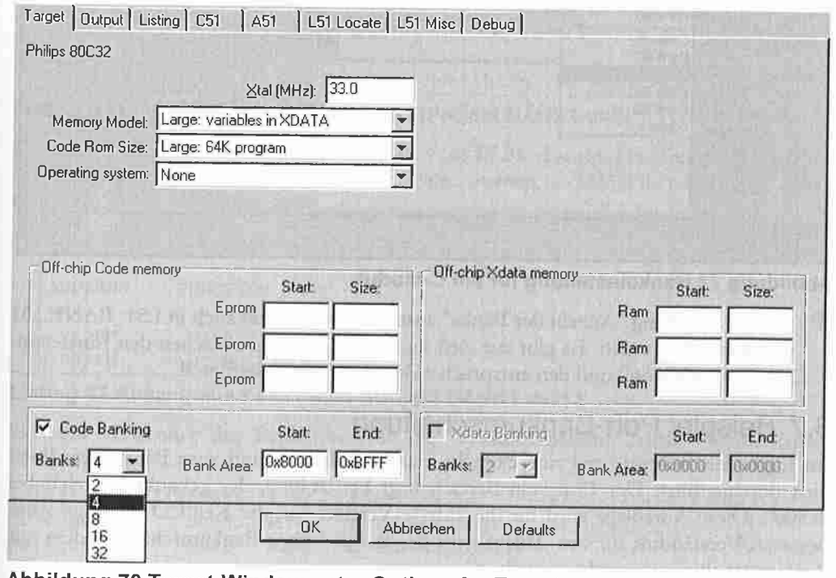


Abbildung 70 Target-Window unter Options for Target

☞ !! Die Anzahl der eingestellten Bänke wirkt sich nur auf das Projekthandling (welches C-Modul muß in welche Bank) aus. Das File L51_BANK.a51 **muß** zusätzlich angepaßt werden. !!

Die unter Abbildung 70 eingetragenen Informationen werden als Input des Linkers übernommen (siehe Abbildung 71).

Abbildung 71 Linker-Direktive mit Banking

Die C-Module können nun selektiv den einzelnen Bereichen zugewiesen werden. Die Auswahl der Bank findet über das Codefenster im Reiter Properties statt (siehe Abbildung 72). Sie erreichen dieses Fenster im Project-Window, indem Sie mit der Maus auf ein Source-Modul gehen und dann die rechte Maustaste betätigen.

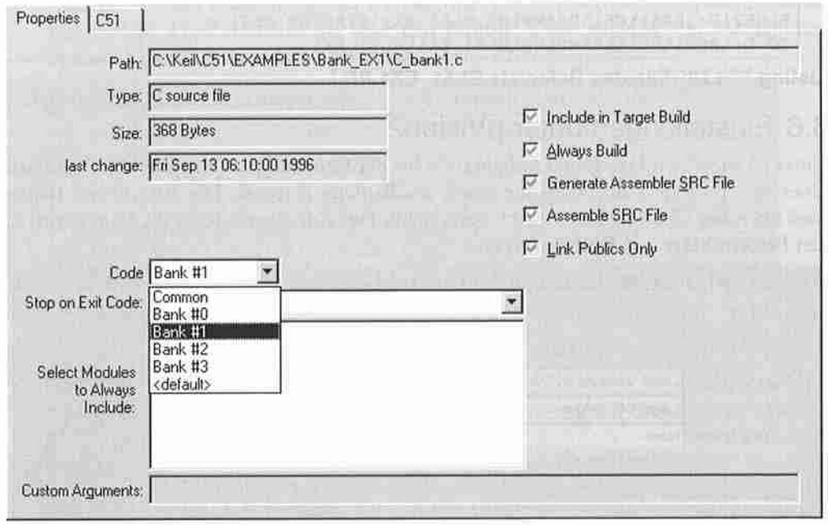


Abbildung 72 Bankeinstellung für ein C-Modul

!! Die Einstellung "Anzahl der Bänke" **muß** im Projekt und auch in L51_BANK.a51 durchgeführt werden. Es gibt zur Zeit keine Verknüpfung zwischen den Einstellungen unter µVision2 und den entsprechenden C-/ Assemblerfiles. !!

6.7 Beispiel Port-Bankumschaltung

Im folgenden Beispiel wird eine einfache Bankumschaltung mit zwei Bänken via Portpins durchgeführt. Der Common-Bereich liegt bei 0x4000, der gebankte Bereich bei 0x5000. Diese Adreßlage wird für die 2kByte Version von der Keil CD benötigt. Zum besseren Verständnis für den Ablauf wird das Beispiel ohne Bankumschaltung dem mit Bankumschaltung gegenübergestellt.

```
   55: main(){
   56:  uint uiErgebnis;
   57:
   58:  uiErgebnis = ui_AddValuesBK0(2,3);
C:0x002D    7D03   MOV    R5,#0x03
C:0x002F    7F02   MOV    R7,#0x02
①  C:0x0031    1156   ACALL  ui_AddValuesBK0(C:0056)
C:0x0033    8E08   MOV    uiErgebnis(0x08),R6
C:0x0035    8F09   MOV    0x09,R7
   59:  uiErgebnis = uiMulValuesBK1(1,2);
C:0x0037    7D02   MOV    R5,#0x02
```



```
C:0x0039    7F01   MOV    R7,#0x01
①  C:0x003B    114E   ACALL  ui_MulValuesBK1(C:004E)
C:0x003D    8E08   MOV    uiErgebnis(0x08),R6
C:0x003F    8F09   MOV    0x09,R7
   60: }
C:0x0041    22     RET
```

Listing 80 Aufruf von Funktionen ohne Banking (Projekt BANK1.uv2))

Listing 80 enthält die disassemblierte C-Funktion main(). Die Funktionen werden direkt mittels ACALL-Aufrufen durchgeführt (siehe Listing 80, ①).

Für die Bankumschaltung muß das File L51_bank.a51 angepaßt und zum Projekt hinzugefügt werden. Die Anpassung der Bankanzahl erfolgt über ?B_NBANKS (siehe Listing 81, ①). Die Bankanzahl wird auf 2 eingestellt.

```
①  ?B_NBANKS  EQU  2  ; Define max. number of banks
   ;                  ; The following values are allowed: 2,4,8,16,32
   ;                  ; the max. value for ?B_BANKS is 32
   ;
   ?B_MODE    EQU  0  ; 0 for Bank-Switching via 8051 Port
```

Listing 81 Einstellung im L51_bank.a51 (Projekt BANK1.uv2)

Die Tabelle für die Bankumschaltung wird vom BL51 erzeugt. Die Einsprungadressen derjenigen Funktionen, die in einer Bank liegen, werden im M51-File ausgegeben (siehe ①).

```
①  INTRABANK CALL TABLE OF MODULE:  bank1 (ADDBK0)

   ADDRESS    FUNCTION NAME
   -----------------------
    002CH     _UI_ADDVALUESBK0
    0031H     _UI_MULVALUESBK1
```

Listing 82 Auszug aus bank1.m51 (Projekt BANK1.uv2))

Generell verwendet der Banklinker für die Bankumschaltung AJMP-Befehle (siehe Listing 83, ③).

```
   ;*********************   Banktabelle   **********************
②  C:0x002C    905000 MOV    DPTR,#ui_AddValuesBK0(0x5000)
③  C:0x002F    0103   AJMP   B_BANK0(C:0003)
④  C:0x0031    905000 MOV    DPTR,#ui_MulValuesBK1(0x5000)
④  C:0x0034    0114   AJMP   B_BANK1(C:0014)
   C:0x0036
     55: main(){
     56:  uint uiErgebnis;
     57:
     58:  uiErgebnis = ui_AddValuesBK0(2,3);
   C:0x0037    7D03   MOV    R5,#0x03
   C:0x0039    7F02   MOV    R7,#0x02
①  C:0x003B    1156   ACALL  C:002C
   C:0x003D    8E08   MOV    uiErgebnis(0x08),R6
   C:0x003F    8F09   MOV    0x09,R7
```

```
  59:  uiErgebnis = ui_MulValuesBK1(1,2);
C:0x0041    7D02    MOV     R5,#0x02
C:0x0043    7F01    MOV     R7,#0x01
C:0x0045    114E    ACALL   C:0031
C:0x0047    8E08    MOV     uiErgebnis(0x08),R6
C:0x0049    8F09    MOV     0x09,R7
  60: }
C:0x004B    22      RET
```

Listing 83 Aufrufe mittels Banking

```
;**********************    Banktabelle    **********************
C:0x002C    905000 MOV     DPTR,#ui_AddValuesBK0(0x5000)
C:0x002F    020003 LJMP    B_BANK0(C:0003)
C:0x0032    905000 MOV     DPTR,#ui_MulValuesBK1(0x5000)
C:0x0035    020014 LJMP    B_BANK1(C:0014)
```

Listing 84 Einsprungtabelle mit der Direktive NOAJMP

Im Gegensatz zur ungebankten Version wird nicht die Adresse der Funktion, sondern die Umschaltfunktion aufgerufen (siehe Listing 83, ①). Sie enthält die absolute Adresse der Funktion sowie einen Sprung zur Umschaltung der Bank. Die Umschaltfunktion lädt zuerst die Adresse des Programms (siehe ②) und ruft dann ihrerseits die Bankumschaltung (siehe ③) auf. Für jede Funktion, die in einer Bank liegt, werden 5 Bytes Code in der Banktabelle benötigt (siehe ④). Die Bankumschaltung finden Sie in Listing 85. Der Umschaltcode für die Bank 0 ist an der Adresse C:0x0100, für Bank 1 an Adresse C:0x0108 abgelegt. Die Bankumschaltung sorgt ihrerseits dafür, daß sowohl die aktive Bank (Listing 85, ①), als auch die Einsprungadresse der Funktion auf dem Stack (siehe ②) gesichert werden.

```
C:0x0003    E590    MOV    A,?B_PORT(0x90)
C:0x0005    5408    ANL    A,#BANK1(0x08)
C:0x0007    C0E0    PUSH   ACC(0xE0)
C:0x0009    7401    MOV    A,#MASK(0x01)
C:0x000B    C0E0    PUSH   ACC(0xE0)
C:0x000D    C082    PUSH   DPL(0x82)
C:0x000F    C083    PUSH   DPH(0x83)
C:0x0011    020100 LJMP    B_SWITCH0(C:0100)
C:0x0014    E590    MOV    A,?B_PORT(0x90)
C:0x0016    5408    ANL    A,#BANK1(0x08)
C:0x0018    C0E0    PUSH   ACC(0xE0)
C:0x001A    7401    MOV    A,#MASK(0x01)
C:0x001C    C0E0    PUSH   ACC(0xE0)
C:0x001E    C082    PUSH   DPL(0x82)
C:0x0020    C083    PUSH   DPH(0x83)
C:0x0022    020100 LJMP    B_SWITCH0(C:0100)
C:0x0100    5390F7 ANL     ?B_PORT(0x90),#0xF7
C:0x0103    22      RET
C:0x0104    00      NOP
C:0x0105    00      NOP
C:0x0106    00      NOP
C:0x0107    00      NOP
```



```
C:0x0108    439008 ORL     ?B_PORT(0x90),#BANK1(0x08)
C:0x010B    22     RET
```

Listing 85 Bankumschaltung

6.8 Direkte Bankumschaltung

Es besteht die Möglichkeit, eine direkte Bankumschaltung mit der Funktion Bankswitch() durchzuführen. Die Funktion ist im Modul L51_BANK.a51 enthalten. Mittels dem Übergabeparameter wird die Bank angegeben. Die direkte Umschaltung wird z. B. dann benötigt, wenn in den Bänken Tabellen mit Konstanten vorhanden sind. In Abhängigkeit des Bankumschaltverfahrens wird die entsprechende Source-Sequenz assembliert. Listing 86 enthält die Aufrufsequenz für die Portumschaltung.

!! Die direkte Bankumschaltung kann nur dann eingesetzt werden, wenn sich in den Bänken Konstanten befinden. Ansonsten kann es zu einem undefinierten Verhalten des Programms kommen. !!

Die aktive Bank wird bei der Bankumschaltung mittels XDATA-Port in der Variable **?B_CURRENTBANK** gespeichert. Die Auswertung der Variablen muß über eine Ausleseroutine erfolgen. Listing 87 enthält die Ausleseroutine für die Banknummer. Bei der Portumschaltung wird die Variable nicht angelegt, da der Port direkt ausgelesen werden kann. Bei der Auswertung der Banknummer muß darauf geachtet werden, daß die nicht verwendeten Portpins bei der Auswertung über eine Und-Verknüpfung ausgeblendet werden.

```
① ;***************  SWITCHBANK FUNCTION  *************************
  ; SWITCHBANK allows use of bank-switching for C programs          *
  ; prototype:   extern switchbank (unsigned char bank_number);    *
  ;*****************************************************************
    PUBLIC  _SWITCHBANK

  _SWITCHBANK:    MOV     A,R7
  IF  ?B_MODE = 0
  ;*****************************************************************

  _SWITCHBANK_A:  SWAP    A
                  RR      A
                  MOV     DPTR,#?BANK?SWITCH
                  JMP     @A+DPTR

  ENDIF  ; close block IF ?B_MODE = 0  ****************************
```

Listing 86 Umschaltsequenz Bankumschaltung (Auszug aus L51_BANK.a51)

```
①       RSEG  ?PR?uc_getBankNumber?BANK_INFO
uc_getBankNumber:
        USING  0
                MOV     R7,?B_CURRENTBANK
               RET
```

Listing 87 Auslesen der Banknummer

6.9 Beispiel XDATA-Bankumschaltung mit Funktionsaufruf

Für den Test wurde das Projekt Bank_ex1.prj unter **\c51\examples\bank_ex1** verwendet. Zum besseren Verständnis wurden die printf()-Anweisungen auskommentiert.

```
?B_NBANKS     EQU  4       ; Define max. Number of Banks          *
?B_MODE       EQU  1       ; 0 for Bank-Switching via 8051 Port   *
?B_XDATAPORT  EQU  0FFFFH  ; default is XDATA Port Address 0FFFFH  *
?B_FIRSTBIT   EQU  0       ; default is Bit 0                     *
```

Listing 88 Einstellung in L51_BANK.a51

In Listing 89 befindet sich das Speicherlayout des gebankten Beispiels. Die Funktionen **?BANK?SELECT** und **?BANK?SWITCH** (siehe Listing 89 ① und ②) sind im File **L51_BANK.a51** enthalten.

```
        * * * * * * *   C O D E   M E M O R Y   * * * * * * *
        CODE     0000H     0003H     ABSOLUTE
        CODE     0003H     008CH     UNIT          ?C_C51STARTUP
①       CODE     008FH     005EH     INBLOCK       ?BANK?SELECT
        CODE     00EDH     0008H     UNIT          ?PR?MAIN?C_ROOT
        CODE     00F5H     0008H     UNIT          ?C_INITSEG
                 00FDH     0003H                   *** GAP ***
②       CODE     0100H     003BH     PAGE          ?BANK?SWITCH

        * * * * * * *   C O D E   B A N K   0   * * * * * * *
                 0000H     8000H                   *** GAP ***
        BANK0    8000H     0003H     UNIT          PR?FUNC0?C_BANK0

        * * * * * * *   C O D E   B A N K   1   * * * * * * *
                 0000H     8000H                   *** GAP ***
        BANK1    8000H     0003H     UNIT          ?PR?FUNC1?C_BANK1

        * * * * * * *   C O D E   B A N K   2   * * * * * * *
                 0000H     8000H                   *** GAP ***
        BANK2    8000H     0001H     UNIT          ?PR?FUNC2?C_BANK2
```

Listing 89 Auszug aus Bank_ex1.m51

Soll ein Sprung in eine Bank durchgeführt werden, wird ein Aufruf in das Select-Modul durchgeführt. In diesem Modul wird die Startadresse der aufgerufenen Funktion in den DPTR geladen (siehe Listing 90, ①,② und ③). Danach wird ein Sprung zur Bankumschaltung durchgeführt.

```
  func0();                          // Funktion liegt in Bank 0
C:0x05BB  1204C7  LCALL   C:00DD    // Sprung nach ?BANK?SELECT
  func1();                          // Funktion liegt in Bank 1
C:0x05BE  1204CC  LCALL   C:00E2    // Sprung nach ?BANK?SELECT
```

```
_SWITCHBANK:
C:0x00D7  EF      MOV     A, R7
C:0x00D8  C4      SWAP    A
C:0x00D9  900100  MOV     DPTR,#B_SWITCH0(0x100)
C:0x00DC  73      JMP     @A+DPTR
```



```
①  C:0x00DD  908000  MOV   DPTR,#func1(0x8000)
   C:0x00E0  018F    AJMP  B_BANK0(C:008F)
②  C:0x00E2  908000  MOV   DPTR,#func1(0x8000)
   C:0x00E5  01A1    AJMP  B_BANK1(C:00A1)
③  C:0x00E7  908000  MOV   DPTR,#func1(0x8000)
   C:0x00EA  01B3    AJMP  B_BANK2(C:00B3)
```

Listing 90 Auszug aus C_ROOT.c

Die Funktion **Bank** bildet die Einsprungadresse für den Rücksprung und sichert diese auf den Stack (siehe Listing 91, ①). Danach wird die Einsprungadresse für die aufgerufene Funktion auf den Stack gelegt (siehe Listing 91, ②).

```
   C:0x008F  E508    MOV   A,?B_CURRENTBANK(0x08)
   C:0x0091  5403    ANL   A,#BANK3(0x03)
   C:0x0093  C4      SWAP  A
①  C:0x0094  C0E0    PUSH  ACC(0xE0)
   C:0x0096  7401    MOV   A,#BANK1(0x01)
①  C:0x0098  C0E0    PUSH  ACC(0xE0)
②  C:0x009A  C082    PUSH  DPL(0x82)
②  C:0x009C  C083    PUSH  DPH(0x83)
   C:0x009E  020100  LJMP  B_SWITCH0(C:0100)
```

Listing 91 Auszug aus ?Bank?Select

Die Funktion **B_SWITCH0** (siehe Listing 92) lädt den Wert zur Umschaltung (in diesem Beispiel durch das Schreiben auf die Adresse 0xFFFF). Zudem wird die Variable **B_CURRENTBANK** mit diesem Wert geladen. Der Aufruf der in der Bank stehenden Funktion erfolgt nun über den RET-Befehl (siehe Listing 92, ①), da auf dem Stack die Einsprungadresse abgespeichert wurde (siehe Listing 91, ②).

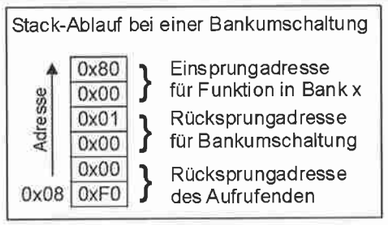
```
②  C:0x0100  F8      MOV   R0,A
   C:0x0101  7400    MOV   A,#?BANK?SWITCHING(0x00)
   C:0x0103  90FFFF  MOV   DPTR,#?B_XDATAPORT(0xFFFF)
   C:0x0106  F508    MOV   ?B_CURRENTBANK(0x08),A
   C:0x0108  F0      MOVX  @DPTR,A
   C:0x0109  E8      MOV   A,R0
①  C:0x010A  22      RET
```

Listing 92 Auszug aus ?BANK?SWITCH für Bank 0

Für jede Bank wird eine Codesequenz, wie in Listing 92 abgebildet, aufgebaut. Über die Startadresse der Bankumschaltung (LSB) kann die zu selektierende Bank ermittelt werden. In Listing 92 (siehe ②) ist die Bankumschaltung für Bank 0 abgebildet.

Wurde die Funktion in der Bank durchlaufen, wird anschließend über den RET-Befehl der aufgerufenen Funktion der Bank-Switch-Mechanismus wieder gestartet, der seinerseits die alte Bank wieder einstellt.

☞ !! Dies wird benötigt, wenn eine Funktion in einer Bank eine weitere Funktion aufruft, die jedoch in einer anderen Bank abgelegt ist. !!



In Abbildung 73 ist die Stackbelegung bei einer Bankumschaltung nochmals dargestellt. Werden Funktionen aus Bänken heraus aufgerufen, die ihrerseits in Bänken liegen, so wird für jede weitere Bankumschaltung 2 Byte Stack benötigt.

Abbildung 73 Stackbelegung bei Aufruf einer Funktion über Bankumschaltung

6.10 Eigene Bankumschaltung

Mit der Version 2.0 des l51_bank.51 besteht nun die Möglichkeit, eine eigene Bankumschaltsequenz zu definieren. Dies wird z. B. dann benötigt, wenn die Bankumschaltung mittels einem EPLD oder FPGA realisiert ist und die Umschaltung über eine Sequenz von Ansteuerbefehlen erfolgt. Über die Definition ?B_Mode EQU 4 (siehe Listing 93, ①) wird die selbst definierte Umschaltsequenz assembliert.

```
  ; Version 2.03(Code and Variable Banking for 8051 based Derivates)
  ;------------------------------------------------------------------
  ;************** Configuration Section ****************************
  ;                                                                  *
① ?B_MODE    EQU  4 ; 0 for Bank-Switching via 8051 Port           *
  ;                   ; 1 for Bank-Switching via XDATA Port         *
  ;                   ; 4 for user-provided bank switch code       *
```

Listing 93 Auszug aus l51_bank.a51

Für die eigene Bankumschaltung ist ein Bereich im l51_bank.a51 vorgesehen. Dieser Bereich wird über die Abfrage ?B_MODE = 4 eingeleitet (siehe Listing 94, ①). Für jede Bank muß eine eigene Bankumschaltung in einem Makro definiert werden (siehe ②, ③).

```
①  IF   ?B_MODE = 4;                                                *
   ;-----------------------------------------------------------------*
   ; if ?BANK?MODE is 4 define the following switch macros          *
   ; For bank switching via user-provided bank switch code you must *
   ; define for each memory bank a own macro which contains the bank*
   ; switch code. The number of macros must conform with the number *
   ; ?B_NBANKS number, i.e. for an application with 16 memory banks *
   ; you must define 16 macros.                                     *
   ;                                                                *
   ; IMPORTANT NOTES:                                               *
   ; 1.  Each SWITCH macro must generate the same code size, other- *
   ;     wise the BANKSWITCH code does not work correctly.          *
   ; 2.  You must ensure that your bankswitch logic has a defined   *
   ;     stated after CPU reset. We recommend that the following    *
   ;     code is inserted at the end of the STARTUP.a51 file:       *
   ;                 :                                              *
   ;     EXTRN DATA (?B_CURRENTBANK)                                *
   ;      MOV     ?B_CURRENTBANK,#0 ; select code bank 0            *
   ;      SWITCH0                   ; the code of your SWITCH0 macro*
```



```
   ;      LJMP     ?C_START                                         *
   ;                  :                                            *
   ; 3.  If you are using the RTX-51 real-time operating system this*
   ;     banking mode cannot be used.                               *
   ;                                                                *
   P1               DATA     90H       ; I/O Port Address          *
   P3               DATA    0B0H       ;                           *
②  SWITCH0          MACRO               ; Switch to Memory Bank #0  *
②                   CLR      P1.2       ; Clear Port 1 Bit 2        *
②                   CLR      P3.4       ; Clear Port 3 Bit 4        *
②                   ENDM
   ;                                                                *
③  SWITCH1          MACRO               ; Switch to Memory Bank #0  *
③                   SETB     P1.2       ; Set   Port 1 Bit 2        *
③                   CLR      P3.4       ; Clear Port 3 Bit 4        *
                    ENDM
   ;----------------------------------------------------------------*
   ENDIF;                                                           *
```

Listing 94 Auszug aus L51_bank.a51

Folgende Regeln müssen bei Verwendung der Bankumschaltung beachtet werden:

1. Die Bankumschaltsequenz muß für jedes Makro die gleiche Codelänge haben.
2. Nach Reset muß die Bankumschaltung in einem definierten Status sein. Ansonsten muß eine Bankumschaltung in der Startup.a51 erfolgen. In diesem Fall muß das Modul startup.obj im Common-Bereich liegen.
3. Sie können kein RTX-51 einsetzen.

6.11 Banking von Assembler-Sourcen

Sollen Assembler-Sourcen für das Banking verwendet werden, müssen diese nach folgenden Regeln aufgebaut sein:

1. Die Segmente müssen mit ?PR? gekennzeichnet sein (siehe ①).
2. Die Parameterübergabe an andere Aufrufe darf nicht über die Register A, B und DPTR erfolgen.

```
①  ?PR?Funktionsname1?Modulname        SEGMENT CODE
①  ?PR?Funktionsname2?Modulname        SEGMENT CODE
   ?DT?Funktionsname1?Modulname        SEGMENT DATA OVERLAYABLE

         PUBLIC Funktionsname1     ; Falls noetig
         PUBLIC Funktionsname2     ; Falls noetig

         RSEG  ?DT?Funktionsname1?Modulname
?Funktionsname1?BYTE:
   iAddress?040:   DS   2
        vPtr?041:   DS   3
         ORG  5
       ucErg?042:   DS   1
```

```
        RSEG  ?PR?Funktionsname1?Modulname
Funktionsname1:
        USING  0
        ...;
       RET

        RSEG  ?PR?Funktionsname2?Modulname
Funktionsname2:
        USING  0
        ...;
       RET
```

Listing 95 Aufbau eines Assemblermoduls

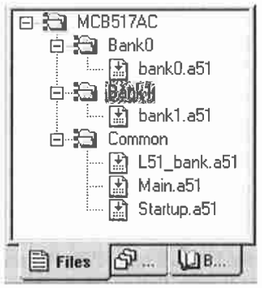


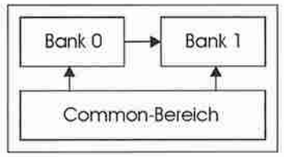
Abbildung 74 enthält das Projekt BANK_ASM.uv2). Für den Start des Progamms wird Startup.a51 verwendet.

☞ !! Die Sprungadresse ?C_START muß in der startup.a51 angepaßt werden, da ein Startmodul mit dieser Adresse nur bei C-Sourcen mit angelegt wird. !!

Abbildung 74 Projekt BANK_ASM.uv2

6.12 Regeln für das Banking

Je weniger Bänke Sie verwenden, desto effektiver und kleiner bleibt der Overhead für den Umschaltmechanismus. Werden nur zwei Bänke benötigt, kann eine zusätzliche Umschaltung eingespart werden, allerdings nur, wenn Sie sicherstellen können, daß in der Bank 1 keine Funktionen enthalten sind, die ihrerseits Funktionen aus Bank 0 benötigen. Sie müssen nur gewährleisten, daß die Bank 0 nach Reset sofort aktiviert wird.



Alle Funktionen, die in dieser Bank stehen, in diesem Beispiel Bank 0, können somit als Common-Funktionen angesehen werden, da sie aus dem Common-Bereich ohne Bankumschaltung aufgerufen werden können (siehe Kapitel 6.15, Seite 130).

Abbildung 75 vereinfachtes Banking

Damit der Umschaltmechanismus klein bleibt, sollten Sie ajmp-Befehle für die Sprünge innerhalb der Umschaltsequenz verwenden.

Versuchen Sie, die am häufigsten verwendeten Funktionen in den Common-Bereich zu legen. Welche Funktionen wie oft aufgerufen werden, können Sie dem Source-Browser entnehmen.

6.13 STARTUP.a51 in ein Banksegment verlagern

Sie haben die Möglichkeit, den Code der startup.a51 in eine Bank zu verlagern. Diese Datei läßt sich **nicht** direkt über die Eingabe der Banknummer in ein Banksegement legen. Um dennoch eine Verlagerung durchzuführen, müssen Sie unter dem Menüpunkt **Options \BL51 Code Banking Linker.**. (siehe Abbildung 76 und Abbildung 77) folgenden Eintrag angeben:

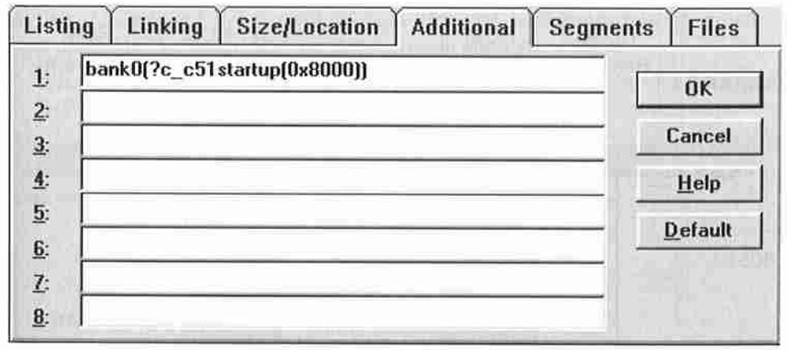


Abbildung 76 Eingabe in µVision Options\BL51..\Addional

☞ !! Die STARTUP.a51 **muß** in eine Bank verlegt werden, die nach dem Reset aktiv ist. !!

Der BL51 wird das Codesegment in die angegebene Bank, in diesem Fall Bank 0, legen. Der Common-Bereich wird als GAP ausgewiesen (siehe Listing 96, ①).

Abbildung 77 Angabe in µVision2 Options for Target\L51 Misc

```
① * * * * * * *   C O D E   B A N K   0   * * * * * * *
          0000H     8000H                  *** GAP ***
  BANK0   8000H     000FH     UNIT         ?C_C51STARTUP
```

Listing 96 Auszug aus M51-File

Dieser Einsprungpunkt ist auch die Ursache für die Warnung des BL51-Linkers.

```
*** WARNING 17: INTERRUPT FUNCTION IN BANKS NOT ALLOWED
    SYMBOL:   ?C_C51STARTUP
    SPACE:    BANK0
```

Listing 97 Ausgabe einer Warnung beim BL51

☞ !! Der BL51 definiert den Reseteinsprung auch als Interrupt-Einsprung. Wollen Sie die Warnung ignorieren, können Sie dies über den Menüpunkt (**Options\Make Options\Misc Ignore Warnings** für µVision) einstellen. !!

6.14 Erweiterter Common-Bereich

Ist der benötigte Common-Bereich größer als der verfügbare Bereich, wird die Differenz automatisch vom Linker in jede Bank eingebracht (siehe Abbildung 78).

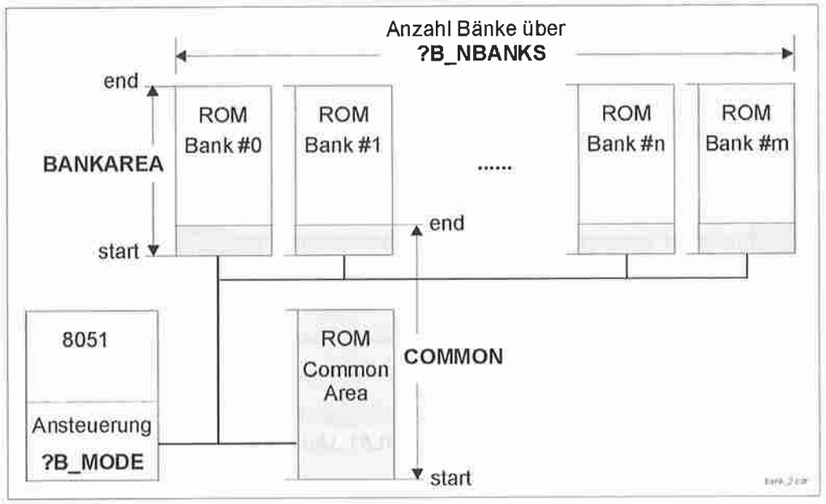


Abbildung 78 Erweiteter Common-Bereich

Es besteht die Möglichkeit die Bankarea an eine beliebige Stelle im Adreßraum des 8051 zu plazieren. So kann z. B. auch die Bankarea zwischen zwei Common-Bereichen liegen.

6.15 Vereinfachtes Banking

Benötigen Sie nur zwei Bänke, können Sie ein einfaches Banking dann anwenden, wenn die Funktionen aus Bank 0 keine Funktionen aus Bank 1 aufrufen. Somit wird nur eine Sprungtabelle für die Funktionen aus Bank1 erzeugt. Dieses Verfahren wird allerdings **nicht** direkt vom BL51 unterstützt. Dies bedeutet, daß Bank 1 mit Ihrem Code auch nicht im Projekt existiert. Damit die Umschaltung mit der Adreßangabe dennoch korrekt arbeitet, wird ein File zum Projekt hinzugebunden, in dem die Einsprungadressen für die jeweiligen Adressen stehen. Es hört sich komplizierter an, als es ist. Dennoch muß einiges bei der Adreßlage beachtet werden. In Abbildung 79 sind die zulässigen bzw. nicht zulässigen Aufrufe zwischen den einzelnen Bänken bzw. dem Common-Bereich dargestellt.

Vorteile

1. Für die Bank 0 wird keine Bankumschaltung generiert. Somit werden für jede Funktion 5 Byte für die Einsprungadresse eingespart.
2. Die Ausführungszeit wird schneller, da die Umschaltsequenz nicht angesprungen wird.
3. Der Stack wird um 2 Byte entlastet (Rücksprungadresse Bankumschaltung).

Nachteile

1. Es müssen zwei Projekte erstellt, compiliert und gelinkt werden.
2. Für den gebankten Bereich muß ein Assemblerfile erstellt werden, das die Einsprungadressen für die jeweiligen Funktionen enthält.
3. Werden Funktionen aus dem Common-Bereich benötigt, so muß auch hier ein Assemblerfile mit den Einsprungadressen erstellt werden.
4. Bei Änderungen des Source können sich die Einsprungadressen ändern. Somit müssen die Assemblerfiles mit den Adressen angepaßt werden.
5. Ein Debuggen auf Sourcelevel ist im gebankten Teil nicht möglich, da dieser Teil als HEX-File zum eigentlichen Projekt hinzugeladen wird.

Die Nachteile sind nicht ganz unerheblich. Dennoch ist dieses Verfahren dann sehr interessant, wenn der ROM- bzw. EPROM-Bereich begrenzt ist oder die Geschwindigkeit eine wesentliche Rolle spielt.

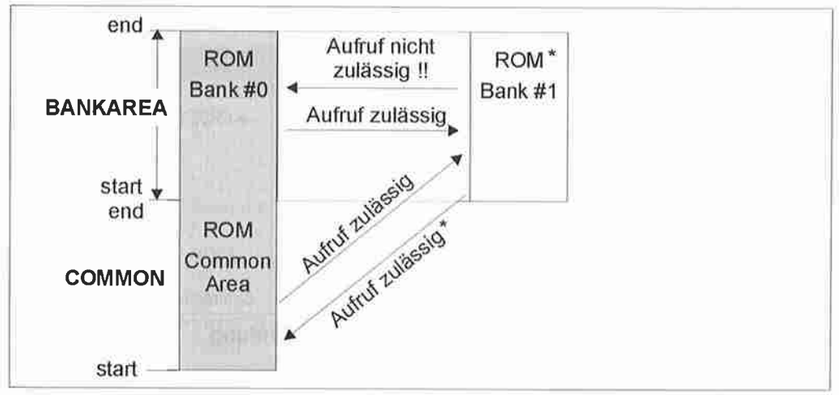


Abbildung 79 Vereinfachtes Banking

Im Projekt BANK2.uv2) kommt das vereinfachte Banking zur Anwendung.

6.16 Arbeiten mit dem OC51 (Banked Object File Converter)

Allgemeines

Der OC51 ist eine Anwendung, die gebankte OBJ-Files in absolute OBJ-Files übersetzt. Der BL51 generiert ein spezielles OBJ-File, wenn ein Programm eine Bankumschaltung enthält. Das gebankte OBJ-File enthält mehrere Codebänke, die an der gleichen physikalischen Adresse starten. Aus diesem Grund ist dieses OBJ-File nicht kompatibel zum OMF-51 Format. Sie müssen den OC51 einsetzen, um ein gebanktes OBJ-File in ein oder mehrere absolute OBJ-Files zu konvertieren. Der OC51 erzeugt für jede Bank ein einzelnes absolutes OBJ-File. Die symbolische Debuginformation, die im gebankten OBJ-File vorhanden ist, wird zum jeweiligen absoluten OBJ-File hinzugefügt. Um aus den absoluten OBJ-Files HEX-Files zu erstellen, wird der OH51 benötigt. Abbildung 80 enthält den Ablauf der Tools für die Erstellung einer gebankten Anwendung. Der Simu-

lator kann mit dem Output des BL51 und mit dem Output des OC51 arbeiten. Wird der Output des OC51 für den Simulator verwendet, so müssen mehrere Files geladen werden, in diesem Beispiel bank1.B00 und bank1.B01. Sie können sich das Laden erleichtern, wenn Sie die Ladekommandos für die zwei Files in ein INI-File eintragen.

```
load bank1.b00  // Laden der Bank 0
load bank1.b01  // Laden der Bank 1
g, main         // Programm laeuft bis zur Codeadresse main
```

Listing 98 Loadbank.ini

☞ !! Werden gebankte Files in den Simulator geladen, sind nur die Symbole der letzten geladenen Bank vorhanden. Somit ist ein Debuggen nur eingeschränkt möglich. !!

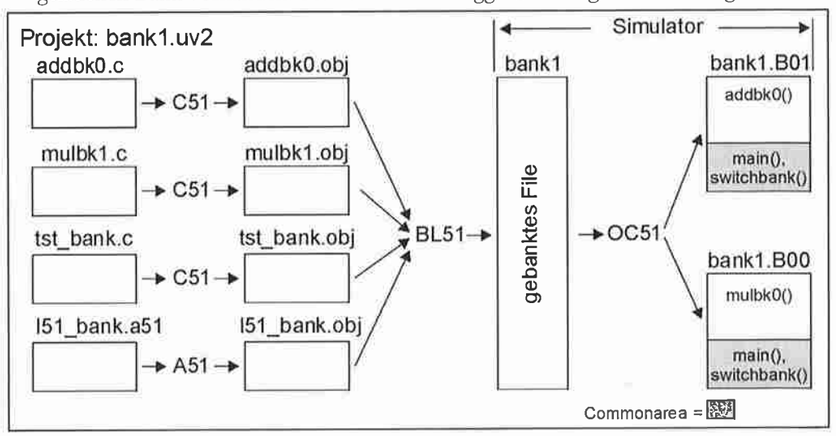


Abbildung 80 Ablauf Erstellung einer gebankten Anwendung

Aufruf des OC51

Der OC51 wird unter µVision2 automatisch gestartet, wenn im Reiter Options for Target\Output die Box Create HEX File aktiviert wird. Der OC51 erzeugt je nach eingestellter Bankanzahl 2 – 32 Files mit den Fileextensions .B00 bis .B31. Im Output Window erscheint für das Projekt BANK1.uv2) folgende Meldung:

```
Aufruf aus der DOS-Box:
  OC51 bank1

BANKED OBJECT FILE CONVERTER OC51 V3.01
COPYRIGHT KEIL ELEKTRONIK GmbH 1991 - 2000
GENERATING STANDARD (UN-BANKED) OBJECT FILES
  OBJECT FILE FOR BANK 0:   BANK1.B00
  OBJEKT FILE FOR BANK 1:   BANK1.B01
creating hex file from "bank1.B00"...
creating hex file from "bank1.B01"...
"bank1" - 0 Error(s), 0 Warning(s).
```

Listing 99 Ausgabe im Output Window bzw. DOS-Box beim Projekt bank1

☞ !! Bei dieser Einstellung werden OC51 und OH51 (Bin->Hexconverter) gestartet. !!

☞ !! In jeder Bank befindet sich außer dem Bankanteil auch der Common-Bereich. !!

Es besteht die Möglichkeit, den OC51 aus einer DOS-Box heraus aufzurufen. Als einzige Angabe ist das gebankte OBJ-File zulässig.

OC51 OBJ-Filename

- **OC51**: Die Eingabe des Filenamens muß beim Start des OC51 erfolgen. Eine relative bzw. absolute Pfadangabe ist bei der Angabe des Filenamens zulässig. Der OC51 erstellt für jede Bank ein eigenes File bestehend aus dem OBJ-Filename mit der Extension Bxx. In xx ist die Banknummer abgelegt.
  ☞ !! Der OC51 kann nur gebankte OBJ-Files verarbeiten. !!

6.17 Gebankte Applikationen im Simulator/ Emulator

Der Simulator bekommt alle Informationen für das Banking aus der geladenen Applikation. Nach dem Start des Simulators werden im Disassembly Window die Adressen in Abhängigkeit, der Bankinformationen angezeigt. Die Adresse setzt sich in den gebankten Bereichen aus der Banknummer und der Adresse zusammen (siehe Abbildung 81).

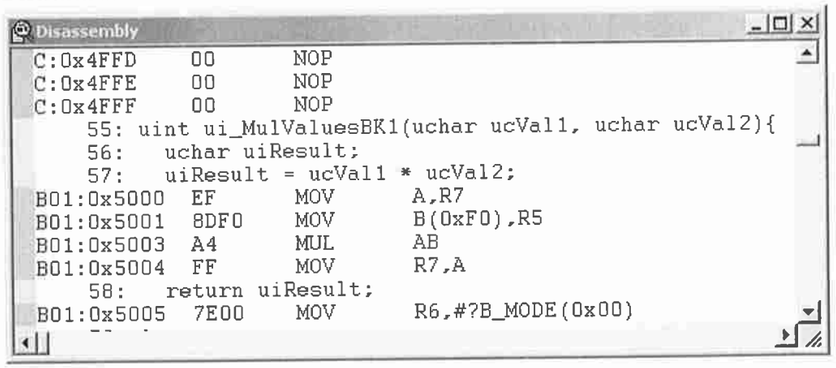


Abbildung 81 Ausgabe in Disassembly Window

Bei der Emulator-Anbindung (Hitex) werden die Bankinformationen direkt aus den Projekteinstellungen von µVision2 entnommen. Diese werden als Grundeinstellung für den Emulator verwendet (siehe auch Kapitel 9.1 Banks, Seite 220).