Externe Programmmodule in 805x-Programmen

Dieter Reiermann, TGM, NT/EL TGM-DSK-169:EXTMOD*.*

Im Schulbetrieb werden meist kleine Programme aus wenigen Zeilen Assemblercode geschrieben. In der Praxis werden Mikroprozessorprogramme, wie man sich leicht vorstellen kann, wesentlich länger. Wenn die Hardware mit externen Speicherbausteinen ausgerüstet ist, wird im Anwendungsbereich der Mikroprozessoren der 805x- Familie nicht unbedingt Speicherplatzmangel herrschen müssen.

Man kann daher etwas verschwenderischer mit dem Speicherplatz umgehen und das Programm streng modular und gut strukturiert entwickeln. Wenn allerdings entgültig die entsprechende (P)ROM-Version eines 805x- Mikroprozessors eingesetzt wird, kann man sich unter Umständen solche allzu "sauberen" Methoden nicht leisten. Es stehen ja dann nur bis zu 8kByte zur Verfügung. Ich möchte im folgenden die Möglichkeiten der modularen Programmentwicklung mit Hilfe der beiden im Schulbetrieb verwendeten 8051-Assembler: ASM51 (Intel) und X8051 (2500AD Software) vergleichen.

Modulare Assemblerprogramme

Was ist eigentlich ein Progammodul?

Ein Programmodul ist ein Programmbaustein. Er kann sowohl Teil eines größeren Programmodules oder Teil des Gesamtprogrammes sein. Er kann in einer größeren Quelldatei stehen (interner Modul) oder eine eigene Quelldatei bilden (externer Modul).

Wozu wird man ein Programm aus externen Modulen aufbauen?

Die einzelnen Bausteine eines Programmes sollen getrennt kodiert und assembliert werden, um nicht zu große Quelldateien entstehen zu lassen (Fehleranfälligkeit).

Wie kann man ein Programm in Module aufteilen?

Programmodule sollen zusammengehörige Funktionen repräsentieren, sie sollen "große funktionelle Stärke" haben.

z.B.: Alle Programmschritte, die zur Funktion "Anzeige" gehören, werden in einem Modul zusammengefaßt ("große Stärke")

z.B.: Alle Eingänge einer Steuerung werden durch ein Modul zusammengefaßt. Hier ist die Funktionalität gering, da nicht alle Eingänge der gleichen Funktion zugehören. Es können Programmschritte nach vielen Kriterien zu Modulen zusammengefaßt werden (gleiche Befehlsschritte, zeitliche oder ablaufbezogene Zusammengehörigkeit, oder, wie beim zweiten Beispiel, datenstrombezogene Zusammengehörigkeit). Nur, wenn die Befehlsschritte eines Moduls zu einem (!) funktionellen Ziel führen, hat dieses Modul hohe funktionale Stärke und ist daher leicht änderbar, austauschbar und wartbar.

Eine weitere Hilfestellung bei der modularen Zerlegung eines Programms bietet das hierarchische Baumdiagramm. Wie in einem Organisationsschema beispielsweise eines Betriebes werden in Kästchen die einzelnen Module in ihrem hierarchischen Bezug zueinander dargestellt [Abb.1]

```
MOD_AB
û--MOD_A
Û--MOD_B
```

Abb.1: Hierarchie der Unterprogramme

Wie können Daten an ein Modul übergeben werden?

Daten werden in Mikroprozessorprogrammen über Speicherplätze übergeben. Man spricht auch von **Parametern**. Grundsätzlich können bestimmte Speicheradressen im RAM oder der Stack als Übergabespeicher dienen. Für den 8051 empfiehlt sich eher, die Parameter in einem Speicherbereich mit ihren Namen (=Adressen) zu übergeben . Die Übergabe über den Stack ist etwas fehleranfälliger, denn dabei werden ja nicht die Adressen, sondern nur eine Startadresse, die Reihenfolge und die Parameteranzahl übergeben. Das führt naturgemäß zu einer höheren Fehleranfälligkeit. Für kleine Programme wird diese Methode auch nicht verwendet.

Assemblieren mit dem ASM51 und RL51

Im folgenden Programmbeispiel sehen Sie die modulare Aufteilung und die Parameterübergabe. Die dabei verwendeten Assemblerdirektiven werden im Beitrag **8051-Assemblerdirektiven** zusammenfassend dargestellt.

Im ersten Assemblerlisting wird ein Rumpfhauptprogramm `Mod_AB` mit den wichtigsten Assemblerdirektiven für die Übergabe von Parametern (Daten und Unterprogrammadressen) dargestellt. Das zweite Modul `Mod_A` zeigt, wie Daten an das Hauptprogramm übergeben werden. Das dritte Modul `Mod_B` empfängt Daten vom aufrufenden Hauptprogramm.

Die drei Module wurden als ASCII-Quelldateien (geschrieben mit WORD 5.0) assembliert:

```
ASM51 extmodx.asm
```

Die daraus entstandenen Objekt-Dateien wurden zu einem Gesamtprogramm gelinkt:

```
RL51 extmod1.obj,extmod2.obj,extmod3.obj
```

Das Ergebnis war eine Binärdatei `extmod1` und eine Linkermap `extmod1.m51`, in der die gesamte Adressbelegung für den gewählten Mikroprozessor zu finden ist.

Was kommt tatsächlich dabei an Maschinencode heraus? Das kann man sich z.B. mit dem Simulator `AVSIM51` anschauen. Allerdings muß vorher aus der Binärdatei eine Intel-HEX-Datei gemacht werden:

```
OH extmod51
```

und dann

```
AVSIM51 ALP'extmod1.HEX'
```

EXTMOD1.LST

```
MCS-51 MACRO ASSEMBLER    MOD_AB                                                                07/02/91   PAGE    1


DOS 4.0 (033-N) MCS-51 MACRO ASSEMBLER, V2.2
OBJECT MODULE PLACED IN EXTMOD1.OBJ
ASSEMBLER INVOKED BY:  D:\8051\ASM51.EXE EXTMOD1.ASM


LOC  OBJ            LINE     SOURCE

                       1     ;Beispiel eines übergeordneten Moduls MOD_AB,
                       2     ;z.Bsp. eines Hauptprogrammes, das von einem
                       3     ;Modul MOD_A Daten bekommt und an ein Modul
                       4     ;MOD_B zur weiteren Bearbeitung übergibt.Die
                       5     ;untergeordneten Module werden über CALL aufgerufen.
                       6     ;VERSION ASM51
                       7     ;================================================
                       8 +1  $TITLE  (Hauptmodul);Name erscheint am Kopf
                       9                         ;jeder Seite des Ausdrucks
                      10     NAME    Mod_AB     ;Name des Moduls
                      11     ;*******************************************
                      12     DataSeg SEGMENT DATA;Interner Datenbereich
----                  13             DSEG AT 50H; Ab Adresse 50H im
                      14                         ;internen RAM definiert
0050                  15     Status: DS 1       ;1 Byte reserviert für
                      16                         ;lokalen Parameter
                      17     ;*******************************************
                      18     DatXSeg SEGMENT XDATA;Externer Datenbereich
                      19     PUBLIC xdat_AB     ;Diese Daten werden in Mod_AB
                      20                         ;definiert
----                  21            XSEG AT 0c000H;Externer Datenbereich
C000                  22     xdat_AB:DS 10     ;Reservierung für 10 Bytes
                      23     EXTRN XDATA(xdat_A);Diese Daten werden im Mod_A
                      24                        ;definiert (ext.Speicher)
                      25     ;*******************************************
                      26     CodeSeg SEGMENT CODE;Bereich für Befehle
                      27     EXTRN   CODE (Mod_A,Mod_B);Adressen von
                      28                               ;Mod_A u. Mod_B
----                  29             CSEG AT 0         ;Beginn des Code-Teils
                      30     ;*******************************************
0030                  31             ORG 30H           ;Start nach den Inter-
                      32                               ;ruptvektoren
0030 120000   F       33     Mod_AB: CALL Mod_A        ;Aufruf von Mod_A
0033 75F00A           34             MOV B,#10         ;Schleifenzähler
0036 900000   F       35     Weiter: MOV DPTR,#xdat_A  ;Zeiger auf Daten
                      36                               ;von Mod_A
0039 E0               37             MOVX A,@DPTR      ;Daten über Akku..
003A 90C000           38             MOV DPTR,#xdat_AB ;Zeiger auf Daten
                      39                               ;von Mod_AB
003D F0               40             MOVX  @DPTR,A     ;..nach xdat_AB
003E D5F0F5           41             DJNZ B,Weiter     ;10 Bytes
0041 120000   F       42             CALL Mod_B        ;Aufruf von Mod_B
0044 7550FF           43             MOV  Status,#0ffH ;auf Addresse Status
                      44                               ;wird 1 Byte geladen
                      45                               ;Selber Name, aber
                      46                               ;andere Adresse als in
                      47                               ;Mod_B
                      48             END

MCS-51 MACRO ASSEMBLER    Hauptmodul                                                            07/02/91   PAGE    2


SYMBOL TABLE LISTING
------ ----- -------


N A M E     T Y P E   V A L U E      A T T R I B U T E S

B. . . . .  D ADDR    00F0H   A
CODESEG. .  C SEG     0000H           REL=UNIT
DATASEG. .  D SEG     0000H           REL=UNIT
DATXSEG. .  X SEG     0000H           REL=UNIT
MOD_A. . .  C ADDR    ----      EXT
MOD_AB . .  C ADDR    0030H   A
MOD_B. . .  C ADDR    ----      EXT
STATUS . .  D ADDR    0050H   A
WEITER . .  C ADDR    0036H   A
XDAT_A . .  X ADDR    ----      EXT
XDAT_AB. .  X ADDR    C000H   A PUB


REGISTER BANK(S) USED: 0

ASSEMBLY COMPLETE, NO ERRORS FOUND
```

EXTMOD2.LST

```
                                1     ;Beispiel eines untergeordneten Moduls Mod_A,
                                2     ;z.Bsp. eines Unterprogrammes, das an ein
                                3     ;Modul übergeornetes Modul Mod_AB Daten abgibt.
                                4     ;VERSION ASM51
                                5     ;=========================================
                                6 +1  $TITLE  (Subr.Mod_A) ;Name erscheint am Kopf
                                7                          ;jeder Seite des Ausdrucks
                                8     NAME    Mod_A        ;Name des Moduls
                                9     ;*****************************************
                               10     DatXSeg SEGMENT XDATA;Externer Datenbereich
                               11     PUBLIC xdat_A        ;Diese Daten werden in Mod_AB
                               12                          ;definiert
----                           13             RSEG DatXSeg ;verschieblicher Speicher-
                               14                          ;bereich für Mod_A
0000                           15     xdat_A: DS 10        ;Reservierung für 10 Bytes
                               16     ;*****************************************
                               17     CodeSeg SEGMENT CODE ;Bereich für Befehle
                               18     PUBLIC Mod_A         ;Startadresse für Mod_A
                               19                          ;hier definiert
----                           20             RSEG CodeSeg ;verschieblicher Codebereich
                               21                          ;für Linker
                               22     ;*****************************************
0000 C0E0                      23     Mod_A:  PUSH ACC
0002 C0D0                      24             PUSH PSW     ;Sichern von Akku u.PSW
0004 C083                      25             PUSH DPH
0006 C082                      26             PUSH DPL     ;Sichern von DPTR
0008 900000   F                27             MOV DPTR,#xdat_A ;Zeiger auf Daten
                               28     ;Hier stehen weitere Befehle zur Erzeugung der
                               29     ;Daten für das Modul Mod_AB
000B D082                      30             POP DPL      ;Zurückholen der
000D D083                      31             POP DPH      ;zwischengespeicherten
000F D0D0                      32             POP PSW      ;Register
0011 D0E0                      33             POP ACC
0013 22                        34             RET
                               35             END
SYMBOL TABLE LISTING
------ ----- -------

N A M E     T Y P E   V A L U E      A T T R I B U T E S

ACC. . . .  D ADDR    00E0H   A
CODESEG. .  C SEG     0014H           REL=UNIT
DATXSEG. .  X SEG     000AH           REL=UNIT
DPH. . . .  D ADDR    0083H   A
DPL. . . .  D ADDR    0082H   A
MOD_A. . .  C ADDR    0000H   R PUB   SEG=CODESEG
PSW. . . .  D ADDR    00D0H   A
XDAT_A . .  X ADDR    0000H   R PUB   SEG=DATXSEG
```

EXTMOD3.LST

```
LOC  OBJ            LINE     SOURCE

                       1     ;Beispiel eines untergeordneten Moduls Mod_B,
                       2     ;z.Bsp. eines Unterprogrammes, das von einem
                       3     ;übergeorneten Modul Mod_AB Daten empf^ngt.
                       4     ;VERSION ASM51
                       5     ;=========================================
                       6 +1  $TITLE  (Subr.Mod_B) ;Name erscheint am Kopf
                       7                          ;jeder Seite des Ausdrucks
                       8     NAME    Mod_B        ;Name des Moduls
                       9     ;*****************************************
                      10     EXTRN XDATA(xdat_AB) ;Daten definiert in Mod_AB
                      11     DataSeg SEGMENT DATA ;Bereich für internen Daten-
                      12                          ;speicher
----                  13             RSEG DataSeg ;verschieblicher Beginn
0000                  14     Status: DS 1         ;1 Byte-Reservierung für
                      15                          ;lokalen Parameter
                      16     ;*****************************************
                      17     CodeSeg SEGMENT CODE ;Bereich für Befehle
                      18     PUBLIC Mod_B         ;Hier ist die Startadresse
                      19                          ;des Mod_B definiert
----                  20             RSEG CodeSeg ;Verschieblich für Linker
                      21     ;*****************************************
0000 C0E0             22     Mod_B:  PUSH ACC
0002 C0D0             23             PUSH PSW     ;Sichern von Akku u.PSW
0004 C083             24             PUSH DPH
0006 C082             25             PUSH DPL     ;Sichern von DPTR
0008 900000   F       26             MOV DPTR,#xdat_AB ;Zeiger auf Daten
                      27     ;Hier stehen weitere Befehle zur Verarbeitung
                      28     ;der von Mod_AB empfangenen Daten
000B 750055   F       29             MOV Status,#01010101B;nicht identisch mit
                      30                          ;Status in Mod_AB
                      31                          ;Adresse Status
000E D082             32             POP DPL      ;Zurückholen der
0010 D083             33             POP DPH      ;zwischengespeicherten
0012 D0D0             34             POP PSW      ;Register
0014 D0E0             35             POP ACC
0016 22               36             RET
                      37             END
SYMBOL TABLE LISTING
------ ----- -------

N A M E     T Y P E   V A L U E      A T T R I B U T E S

ACC. . . .  D ADDR    00E0H   A
CODESEG. .  C SEG     0017H           REL=UNIT
DATASEG. .  D SEG     0001H           REL=UNIT
DPH. . . .  D ADDR    0083H   A
DPL. . . .  D ADDR    0082H   A
MOD_B. . .  C ADDR    0000H   R PUB   SEG=CODESEG
PSW. . . .  D ADDR    00D0H   A
STATUS . .  D ADDR    0000H   R       SEG=DATASEG
XDAT_AB. .  X ADDR    ----      EXT
```

Assemblieren mit X8051 und LINK (mit LIB) von 2500AD

Da Assembler und Linker von 2500AD aus einem C-Compilerpaket kommen, kann besonders gut mit Hilfe von Objektprogramm-Bibliotheken gearbeitet werden. Die Anpassung an den individuellen Prozessor aus der 8051-Familie erscheint mir bei dieser Software etwas schwieriger. Grundsätzlich gibt es die Möglichkeit mit Assembler und Linker ähnlich wie bei Intel zu arbeiten. Vorsicht: MODULE- und ENDMOD-Direktiven veranlassen den Assembler nämlich, ein Objektfile in gepckter Form `*.pak` zu erstellen, das nur vom Librarian, dem Bibliotheksverwaltungsprogramm, gelesen werden kann. Die Listings der drei Module zeigen diese Direktiven mit vorangestellten ";". Absolute Adressangaben werden erst beim Linkerlauf eingegeben.

Zur Assemblierung wird die Kommandozeile

```
X8051 extmodxa -d
```

eingegeben (`-d` für Listing auf Disk). Aus `extmodxa.asm` entstehen `extmodxa.obj,extmodxa.lst`.

Nach Aufruf von LINK wird nach den Objektdateien (Inputfiles) und den zu ihren Speicherbereichen gehörigen Adressoffsets gefragt. Die Optionen des Linkers werden D und für den Test mit dem AVSIM51 H heißen (D für Linker-map, H für Intel-Hex-Format). Der Linker produziert daher ein `extmod1a.hex` und ein `extmod1a.map`.

EXTMOD1A.MAP

```
Global Symbol Name  Global Value Global Filename

Mod_A               0047         extmod2a.obj
Mod_B               005B         extmod3a.obj
xdat_A              C00A         extmod2a.obj
xdat_AB             C000         extmod1a.obj
*********************************************
*          L O A D      M A P                 *
*********************************************
*    Section Name   Startad.  Endad.   Size    *
*********************************************
* extmod1a.obj                                 *
*    CODE           0030      0046     0017    *
* extmod2a.obj                                 *
*    CODE           0047      005A     0014    *
* extmod3a.obj                                 *
*    CODE           005B      0071     0017    *
*********************************************
Linker Output Filename : extmod1a.hex
Disk Listing  Filename : extmod1a.map
Symbol Table  Filename : extmod1a.sym /2500 A.D.
Link Errors :  0      Output Format :  Intel Hex
```

Das Intel-Hex-File extmod1a.hex kann dann direkt in das Simulatorprogramm AVSIM51 geladen werden.

Wenn man das 2500AD Entwicklungssystem noch besser nützen will, kann das Bibliotheksverwaltungsprogramm LIB benützt werden. Bedienung ganz einfach: Zuerst Quelldateien mit Moduldefinition (MODULE und ENDMOD-Direktiven) erzeugen, Assemblieren. Nun entstehen statt `*.obj *.pak` Dateien. LIB aufrufen. Mit NEW neue Bibliotheksdatei erzeugen, mit ADD Module aus den `*.pak` Dateien eintragen. Eventuell mit LIST und STAT die Modulparameter anschauen. Mit `j` und `k` kann man durch die Modulliste scrollen. Mit QUITT oder EXIT aus LIB aussteigen. Im Linker kann nun die Bibliotheksdatei vor dem Linken angegeben werden. Referenzen werden automatisch aufgelöst.

EXTMOD1A.LST

```
 1                                  .TITLE  Hauptmodul
 2                                          ;Name erscheint am Kopf
 3                                          ;jeder Seite des Ausdrucks
 4                                  .COMMENT ;Beispiel eines übergeordneten Moduls
 5                                  MOD_AB,z.Bsp. eines Hauptprogrammes, das von einem
 6                                  Modul MOD_A Daten bekommt und an ein Modul
 7                                  MOD_B zur weiteren Bearbeitung übergibt.Die
 8                                  untergeordneten Module werden über CALL aufgerufen.
 9                                  VERSION X8051;
11
12                                  ;.MODULE  Mod_AB   ;Modulbestimmung für die
13                                          ;Verwendung in "Librarian"-
14                                          ;Bibliothekverwaltungsprg.
15                                  ;*********************************************
16  0000                                    DATA       ;Datenbereich
17  0050                                    ORG 50H    ;Start ab 50H, weitere
18                                          ;Offset kann beim Linken angegeben werden
20  0050                            Status: DS 1       ;1 Byte reserviert für lokalen Parameter
22                                  ;*********************************************
23  0000                            DatXSeg SECTION    ;selbstdef. Speicherbereich
24                                          ;hier Daten, eingebettet in den
25                                          ;vordefinierten Bereich "DATA"
26                                  .PUBLIC xdat_AB     ;Diese Daten werden in Mod_AB
27                                          ;definiert, die Offset 0C000H
28                                          ;wird beim Linken angegeben
29  0000                            xdat_AB:DS 10     ;Reservierung für 10 Bytes
30                                  .EXTERN  xdat_A    ;Diese Daten werden im Mod_A
31                                          ;definiert (ext.Speicher)
32                                          ENDS       ;Ende des selbstdef. Speicher-
33                                          ;bereiches
34                                  ;*********************************************
35  0000                                    CODE       ;vordef. Bereich für Befehle
36                                  .EXTERN  Mod_A,Mod_B
37                                          ;Adressen von Mod_A u. Mod_B
38                                  ;*********************************************
39  0030                                    ORG 30H            ;Start nach den Inter-
40                                                             ;ruptvektoren
41  0030  12 00 00                  Mod_AB: CALL Mod_A         ;Aufruf von Mod_A
42  0033  75 F0 0A                          MOV B,#10          ;Schleifenzähler
43  0036  90 00 00                  Weiter: MOV DPTR,#xdat_A   ;Zeiger auf Daten
44                                                             ;von Mod_A
45  0039  E0                                MOVX A,@DPTR       ;Daten über Akku..
46  003A  90 00 00                          MOV DPTR,#xdat_AB  ;Zeiger auf Daten von Mod_AB
48  003D  F0                                MOVX  @DPTR,A      ;..nach xdat_AB
49  003E  D5 F0 F5                          DJNZ B,Weiter      ;10 Bytes
50  0041  12 00 00                          CALL Mod_B         ;Aufruf von Mod_B
51  0044  75 50 FF                          MOV  <Status,#0ffH ;auf Addresse Status
52                                                             ;wird 1 Byte geladen
53                                                             ;Selber Name, aber
54                                                             ;andere Adresse als in
55                                                             ;Mod_B
56                                   ;      ENDMOD
57  0047                                    END
```

EXTMOD2A.LST

```
 1                                      .TITLE  Subr.Mod_A
 2                                              ;Name erscheint am Kopf
 3                                              ;jeder Seite des Ausdrucks
 4                                      ;Beispiel eines untergeordneten Moduls Mod_A,
 5                                      ;z.Bsp. eines Unterprogrammes, das an ein
 6                                      ;Modul übergeordnetes Modul Mod_AB Daten abgibt.
 7                                      ;VERSION X8051
 8                                      ;=============================================
 9
10                                      ;.MODULE Mod_A       ;Modulbestimmung, für die
11                                              ;Verwendung im "Librarian"-
12                                              ;Bibliotheksverwaltungsprg.
13                                      ;*********************************************
14   0000                               DATA         ;Datenbereich
15   0000                               DatXSeg SECTION      ;selbstdefinierter
16                                              ;Datenbereich, Speicherfest-
17                                              ;legung wird beim Linken an-
18                                              ;gegeben
19                                      .PUBLIC xdat_A       ;Diese Daten werden in Mod_A
20                                              ;definiert
21   0000                               xdat_A: DS 10        ;Reservierung für 10 Bytes
22                                              ENDS         ;Ende des selbstdefinierten
23                                              ;Datenbereiches
24                                      ;*********************************************
25   0000                                       CODE         ;Bereich für Befehle
26                                      .PUBLIC Mod_A        ;Startadresse für Mod_A
27                                              ;wird beim Linken angegeben
28                                      ;*********************************************
29   0000   C0 E0                       Mod_A:  PUSH ACC
30   0002   C0 D0                               PUSH PSW     ;Sichern von Akku u.PSW
31   0004   C0 83                               PUSH DPH
32   0006   C0 82                               PUSH DPL     ;Sichern von DPTR
33   0008   90 00 00                            MOV DPTR,#xdat_A ;Zeiger auf Daten
34                                      ;Hier stehen weitere Befehle zur Erzeugung der
35                                      ;Daten für das Modul Mod_AB
36   000B   D0 82                               POP DPL      ;Zurückholen der
37   000D   D0 83                               POP DPH      ;zwischengespeicherten
38   000F   D0 D0                               POP PSW      ;Register
39   0011   D0 E0                               POP ACC
40   0013   22                                 RET
41                                              ;ENDMOD
42   0014                                       END
```

EXTMOD3A.LST

```
 1                                      .TITLE  Subr.Mod_B
 2                                              ;Name erscheint am Kopf
 3                                              ;jeder Seite des Ausdrucks
 4                                      .COMMENT ;Beispiel eines untergeordneten
 5                                      Moduls Mod_B,z.Bsp. eines Unterprogrammes,
 6                                      das von einem übergeordneten Modul Mod_AB
 7                                      Daten empfängt.
 8                                      VERSION ASM51
 9                                      ============================================;
10                                      ;.MODULE Mod_B       ;Modulbestimmung für die
11                                              ;Verwendung im "Librarian"
12                                              ;Bibliotheksverwaltungsprg.
13                                      ;*********************************************
14   0000                                       DATA         ;vordefinierter Speicher-
15                                              ;bereich,Adressfestlegung
16                                              ;beim Linken
17   0000                               DatXSeg SECTION      ;selbstdef. Datenbereich
18                                              ;eingebettet in den vordef.
19                                              ;Speicherbereich "DATA",
20                                              ;Adressfestlegung beim
21                                              ;Linken
22                                      .EXTERN  xdat_AB      ;Daten definiert in Mod_AB
23                                              ENDS         ;Ende des selbstdefinierten
24                                              ;Speicherbereichs
25   0000                               Status: DS 1         ;1 Byte-Reservierung für
26                                              ;lokalen Parameter im
27                                              ;vordefinierten Speicherber.
28                                              ;"DATA"
29                                      ;*********************************************
30   0000                                       CODE         ;vordef.Bereich für Befehle
31                                              ;Offset wird beim Linken an-
32                                              ;gegeben
33                                      .PUBLIC Mod_B         ;Hier ist die Startadresse
34                                              ;des Mod_B definiert
35                                      ;*********************************************
36   0000   C0 E0                       Mod_B:  PUSH ACC
37   0002   C0 D0                               PUSH PSW     ;Sichern von Akku u.PSW
38   0004   C0 83                               PUSH DPH
39   0006   C0 82                               PUSH DPL     ;Sichern von DPTR
40   0008   90 00 00                            MOV DPTR,#xdat_AB ;Zeiger auf Daten
41                                      ;Hier stehen weitere Befehle zur Verarbeitung
42                                      ;der von Mod_AB empfangenen Daten
43   000B   75 00 55                            MOV <Status,#01010101B
44                                              ;nicht identisch mit
45                                              ;Status in Mod_AB
46   000E   D0 82                               POP DPL      ;Zurückholen der
47   0010   D0 83                               POP DPH      ;zwischengespeicherten
48   0012   D0 D0                               POP PSW      ;Register
49   0014   D0 E0                               POP ACC
50   0016   22                                 RET
51                                              ;ENDMOD
52   0017                                       END
```